Universidade de Aveiro Departamento de Matemática
2008

Maria João da Costa Antunes Inácio

Árvore de Suporte de Custo Mínimo com Restrições de Salto

Universidade de Aveiro Departamento de Matemática
2008

Maria João da Costa Antunes Inácio

Árvore de Suporte de Custo Mínimo com Restrições de Salto

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Matemática, realizada sob a orientação científica da Professora Doutora Maria Cristina Saraiva Requejo Agra, Professora Auxiliar do Departamento de Matemática da Universidade de Aveiro.

o júri

presidente

Doutor Domingos Moreira Cardoso
Professor Catedrático da Universidade de Aveiro.

Doutora Maria Adelaide da Cruz Cerveira
Professora Auxiliar da Universidade de Trás-os-Montes e Alto Douro.

Doutora Maria Cristina Saraiva Requejo Agra
Professora Auxiliar da Universidade de Aveiro (orientadora).

agradecimentos

É com muito prazer que dedico esta página a todos aqueles que com o seu carinho e compreensão me deram força e incentivo para a realização deste trabalho. Não quero deixar de agradecer em particular:

à minha orientadora, Professora Doutora Maria Cristina Saraiva Requejo Agra pela sua dedicação a este projecto. Agradeço toda a disponibilidade e apoio que sempre me prestou ao longo da elaboração desta tese, o tema de Investigação proposto e a oportunidade que me deu de estudar sob a sua orientação.

à minha mãe, Luisa, pela alegria e interesse com que viu nascer este projecto e por sempre me ter apoiado.

à minha amiga, Isabel, pelo apoio e coragem que me deu nas fases mais difíceis.

em especial, agradeço ao meu marido, João, por ter estado sempre ao meu lado, a sua comprensão e apoio foram fundamentais na realização deste trabalho.

palavras-chave

Árvore de Suporte de Custo Mínimo, Restrições de Salto, Algoritmo Dual Ascendente.

resumo

Neste trabalho descrevemos um algoritmo Dual Ascendente para o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto (HMST). O problema HMST modela o desenho de uma rede de telecomunicações centralizada com restrições de salto. Estas restrições estão relacionadas com a *performance* da rede, uma vez que limitam o número de ligações que podem ser utilizadas para ligar o computador central a qualquer um dos terminais e garantem uma certa qualidade de serviço no que diz respeito a alguns critérios de *performance* tais como disponibilidade, fiabilidade e tempos de atraso máximo de transmissão. Apresentamos duas formulações de fluxos orientadas já apresentadas para este problema. A primeira obtém-se de uma conhecida formulação de fluxos para o problema da Árvore de Suporte de Custo Mínimo adicionando as restrições de salto e a segunda é uma formulação que usa índices de salto, é mais compacta, e foi obtida por Gouveia utilizando a técnica de redefinição de variáveis de Martin.

Como o problema é $\mathcal{NP}$-*difícil* centrámos a nossa atenção na obtenção de um Algoritmo Dual Ascendente para obter um limite inferior para o valor óptimo deste problema e construímos uma heurística baseada na solução Dual Ascendente que nos permitiu obter um limite superior. A técnica Dual Ascendente consiste, essencialmente, numa forma de resolução do problema dual (ou da relaxação lagrangeana ou da relaxação linear) que tira vantagem da estrutura especial que o problema dual tem. Os resultados computacionais que apresentamos para avaliar a qualidade dos valores obtidos indicam que, apesar do algoritmo Dual Ascendente e da heurística baseada na solução dual ascendente permitirem de uma forma muito rápida obter, respectivamente, um limite inferior e um limite superior para o valor óptimo do problema, estes limites são de fraca qualidade.

keywords

Minimum Spanning Tree Problem, Hop Constrained, Dual Ascent Algorithm.

abstract

In this thesis we describe a Dual Ascent algorithm to the Hop-Constrained Minimum Spanning Tree Problem (HMST). This problem models the design of centralized telecommunication network with hop constraints. These restrictions are related to the network performance. They limit the number of connections that can be used to link the central computer to any of the terminals and they guarantee a certain quality of service with respect to some performance constraints such as availability, reliability and the maximum transmission delay. We present two direct flow formulations already presented for this problem. The first one is obtained from a known flow formulation for the Minimum Spanning Tree Problem adding hop constraints and the second one is a formulation which uses hop-indexes, is more compact and it was obtained by Gouveia using the variable redefinition technique of Martin.

As the problem is $\mathcal{NP}$-*hard* we focus our attention on obtaining a Dual Ascent Algorithm to get to a lower bound to the optimal value of this problem and we build a heuristic based on the dual ascent solution which made it possible to get an upper bound. The Dual Ascent method consists essentially on a way to solve the dual problem (or from the lagrangean relaxation or from the linear relaxation) which takes advantage from the special structure of the dual problem. The computational results we present to evaluate the quality of the obtained values indicate that, although the algorithm Dual Ascent and the heuristic based on the dual ascent solution allow us to obtain in a very rapid way, respectively, a lower and an upper bound for the optimal value of the problem, these bounds are very poor.

Conteúdo

Capítulo 1

Introdução

Neste trabalho trataremos de um problema de Optimização Combinatória denominado por problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto, que designamos por HMST (***H**op-constrained **M**inimum **S**panning **T**ree*). O objectivo deste problema é o de, dado um grafo e um nodo do grafo, que se designa por nodo raíz e habitualmente coincide com o nodo 0, para um parâmetro fixo H, encontrar uma árvore de suporte de custo mínimo com a restrição adicional de que o número de arestas no único caminho desde o nodo raíz até qualquer outro nodo da árvore não é superior a H.

Tal como muitos outros problemas de Optimização Combinatória, o problema HMST é um problema $\mathcal{NP}$*-difícil* e por isso não se conhecem algoritmos que garantam a resolução de todas as suas instâncias em tempo polinomial. No entanto quando $H = 1$ ou H é maior ou igual que o comprimento do maior caminho na árvore de suporte, este problema é de fácil resolução. No primeiro caso a sua solução óptima corresponde a um grafo estrela, no segundo caso as restrições de salto são redundantes e o problema HMST reduz-se ao problema de encontrar a árvore de suporte de custo mínimo. Para este caso especial conhecem-se algoritmos de resolução eficientes que, em tempo polinomial, permitem obter a sua solução óptima como é o caso do Algoritmo de Prim ou do Algoritmo de Kruskal (ver, por exemplo, em Ahuja, Magnanti e Orlin[1]).

Quando $H = 2$ este problema é equivalente a um caso particular do Problema de Localização Simples sem Restrições de Capacidade (SUFL - *Simple Uncapacitated Facility*

Location), se considerarmos que a possível localização dos serviços coincide com a localização dos clientes a serem servidos e associarmos as decisões de abrir um serviço na localização j e servir um cliente i por um serviço localizado em j com a inclusão, respectivamente, dos arcos $(0, j)$ e (j, i) na solução HMST.

O problema SUFL é conhecido como sendo um problema $\mathcal{NP}$-*difícil*[3] o que implica que o problema HMST é também um problema $\mathcal{NP}$-*difícil*.

Referimos ainda que, Manyem e Stallmann[15] provaram que o problema HMST não está em APX, isto é na classe de problemas para os quais é possível ter heurísticas de tempo polinomial com a garantia de obter bons limites inferiores e superiores.

O problema HMST modela o desenho de uma rede de telecomunicações centralizada com restrições adicionais relacionadas com a *performance* (desempenho) da rede. O nodo raíz corresponde ao computador central, os restantes nodos a terminais que têm de estar ligados ao computador central e as arestas correspondem aos potenciais fios de ligação.

As restrições de salto limitam o número de ligações que podem ser utilizadas para ligar o computador central a qualquer um dos terminais, e garantem uma certa qualidade de serviço no que diz respeito a algumas restrições de *performance* tais como disponibilidade[1] *(availability)*, fiabilidade[2] *(reliability)* e tempos de atraso máximo de transmissão.

As restrições de salto modelam tempos de atraso máximo de transmissão na medida em que o tempo de transmissão em qualquer ligação é quase desprezível e como consequência o atraso máximo associado a cada mensagem em qualquer caminho é directamente proporcional ao número de saltos nesse caminho. Assim, caminhos com um comprimento muito grande podem degradar a *performance* da rede, dado que a existência de muitos nodos nos caminhos podem originar tempos de transmissão muito elevados. Restrições de salto modelam também restrições de fiabilidade, porque se estabelecermos um valor baixo para o número máximo de saltos permitidos diminuimos a probablidade de falha em cada caminho. Referimos ainda que (ver Gouveia[10]), desigando por α a fiabilidade associada a cada ligação e considerando que as falhas em cada ligação (i, j) são independentes então a

[1] Probabilidade de todas as linhas de transmissão no caminho do terminal ao computador central estarem a funcionar

[2] Probabilidade de uma sessão não ser interrompida por falha de uma ligação

fiabilidade em cada caminho é dada por α^q, onde q representa o número de ligações nesse caminho. Por forma a garantir que cada mensagem enviada do computador central para cada terminal tem pelo menos uma probabilidade β de chegar ao destino, o que corresponde a garantir que a fiabilidade associada a cada caminho desde o nodo central até qualquer outro nodo terminal não é inferior a β, temos de estabelecer um valor baixo para o número de ligações permitidas. Note que à medida que q aumenta α^q diminui e rapidamente se torna inferior a β. Por exemplo, se considerarmos $\alpha = 0.95$ e $\beta = 0.8$, então não mais do que 4 saltos devem ser incluídos em cada caminho que comece na raíz.

Adicionalmente, restrições de salto podem ser utilizadas para evitar a degradação da qualidade do sinal. Em geral caminhos de voz são limitados a 2 saltos enquanto que caminhos de dados são limitados a 3 saltos.

Resumindo, caminhos com menos saltos originam redes com uma melhor *performance*.

Woolston e Albin[21] apresentaram alguns resultados computacionais que mostram que árvores de suporte sem limites de salto no caminho desde o computador central até aos terminais tem um comportamento insuficiente no que diz respeito a disponibilidade e fiabilidade. Mostraram ainda que, incluindo restrições de salto é possível obter redes com muito melhor *performance* e apenas com um aumento moderado do custo total.

Gouveia[9, 10, 11] apresentou diversas formulações para o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto. Apresentou[9] uma formulação baseada na formulação de eliminação de fluxos de Miller-Tucker-Zemlin. Apresentou e comparou[10] uma formulação de fluxos não orientada com uma formulação de fluxos orientada e mostrou que os valores óptimos das relaxações lineares destas formulações coincidem. Deste modo, e uma vez que o número de restrições é menor no caso orientado, a formulação orientada é mais adequada para utilizar em processos de obtenção de limites inferiores. Neste trabalho do autor são também apresentadas restrições (de *lifting*) para o modelo orientado cuja inclusão no modelo produz melhoramentos no valor do limite inferior. O autor descreve, ainda, esquemas de obtenção de limites inferiores para o valor óptimo do problema baseados em relaxações lagrangeanas combinadas com técnicas de optimização por subgradiente e descreve ainda, uma heurística simples que transforma uma árvore de suporte de custo mínimo numa árvore de suporte com restrições de salto que lhe permite

obter limites superiores para o valor óptimo do problema HMST. Posteriormente [11] construiu uma formulação de fluxos com índices de salto, utilizando a técnica de redefinição de variáveis de Martin[16], e mostrou que se tratava de uma formulação mais compacta, isto é, que permite obter melhores limites inferiores para o problema HMST.

Gouveia e Requejo[12] apresentaram uma nova relaxação lagrangeana baseada nesta última formulação. Os resultados computacionais que obtiveram mostraram que os limites inferiores obtidos pela nova relaxação dominavam os limites inferiores obtidos com as anteriores relaxações lagrangeanas.

Dahl[4] estudou um caso particular deste problema em que $H = 2$. Apresentou uma descrição completa do envolvente convexo do conjunto de restrições do problema quando o grafo subjacente é um ciclo com n nodos e com um nodo raíz ligado a todos os restantes nodos. Apresentou também algumas desigualdades válidas para o caso geral.

A dificuldade de resolução dos problemas $\mathcal{NP}$-*difícil* tem levado os investigadores a procurarem técnicas de resolução que permitam obter bons limites para o valor da solução óptima. Um processo possível, consiste no uso da formulação dual do problema em questão. A técnica Dual Ascendente insere-se nesta classe de técnicas e consiste, essencialmente, numa forma de resolver o problema dual (ou da relaxação lagrangeana ou da relaxação linear) que tira vantagem da estrutura especial que o problema dual tem.

O uso da técnica Dual Ascendente, mais precisamente, a construção de um algoritmo Dual Ascendente para o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto é o objectivo principal deste trabalho.

Como veremos, considerando uma formulação de fluxos orientada para este problema e construindo o problema dual da sua relaxação linear, observamos que fixando o valor de algumas das variáveis duais, o problema se pode separar em vários subproblemas. Cada um desses subproblemas corresponde ao problema dual do problema do caminho mais curto com restrições de salto entre a raíz e o nodo k. Assim, facilmente se encontra uma solução para cada subproblema e consequentemente uma solução dual admissível. É pelo facto do problema assumir esta estrutura que torna tão atractiva a aplicação de um algoritmo Dual Ascendente.

Muito resumidamente, podemos dizer que a estratégia dos algoritmos Duais Ascendentes consiste em começar com uma solução dual admissível e depois iterativamente alterar o valor das variáveis duais de modo a que, em cada iteração, o valor da função objectivo dual melhore. O valor final da solução dual fornece um limite (inferior ou superior) para o valor óptimo do problema. No nosso caso, dado que o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto é um problema de minimização, o valor da solução dual ascendente fornece um limite inferior para o valor óptimo do problema.

Na literatura encontram-se vários trabalhos nos quais esta técnica tem sido aplicada com sucesso (no sentido em que foram obtidos bons resultados computacionais) a vários problemas de optimização em redes. Exemplos são os algoritmos Duais Ascendentes desenvolvidos por Erlenkotter[6] para o Problema de Localização Simples sem Restrições de Capacidade (*Uncapacitated Facility Location*), Wong[20] para o Problema da Árvore de Steiner em Grafos Orientados (*Steiner Tree Problems on a Directed Graph*), Balakrishnan, Magnanti e Wong[3] para o Problema de Desenho de Redes sem Restrições de Capacidade (*Fixed-Charge Uncapacitated Network Design Problems*), Balakrishnan, Magnanti e Mirchandani[2] para o Problema de Desenho de Redes por Níveis (*Multi-level Network Design*) e Rosenberg[19] para um caso especial do Problema de Desenho Redes de Telecomunicações sem Restrições de Capacidade (*Uncapacitated Telecommunications Network Design Problem with Access, Backbone and Switch Costs*). Em todos estes trabalhos a técnica Dual Ascendente foi aplicada ao problema dual da relaxação linear e os testes computacionais realizados pelos autores revelaram que esta técnica fornece, em geral, boas soluções.

Podemos ainda encontrar outros algoritmos Duais Ascendentes para os quais não foram apresentados resultados computacionais, em seguida faz-se referência a alguns desses trabalhos. Herrmann, Ioannou, Minis e Proth [13] tentaram generalizar o algoritmo Dual Ascendente desenvolvido por Balakrishnan et al.[3] para o Problema de Desenho de Redes Com Custos Fixos e Restrições de Capacidade (*Fixed-Charge Capacitated Network Design Problems*). Contudo, Gendron[8] mostrou através de um contra exemplo que o algoritmo

Dual Ascendente proposto por estes autores está incorrecto, no sentido de que não gera um limite inferior válido para o valor óptimo do problema. Gendron, fornece um algoritmo Dual Ascendente válido, baseado na mesma ideia de Herrmann et al.. Pallottino e Scutella[18] desenvolveram um algoritmo Dual Ascendente para o problema da árvore de caminhos mais curto (*Shortest Path Tree Problem*). No seguimento deste trabalho, Nguyen, Pallottino e Scutella [17] desenvolveram um algoritmo Dual Ascendente para o Problema da Reoptimização do Caminho Mais Curto (*Shortest Path Reoptimization Problem*). Ambos os algoritmos referidos foram elaborados com base no problema dual da relaxação linear. Frangioni e Gallo[7] desenvolveram um algoritmo Dual Ascendente para o Problema de Fluxos de Custo Mínimo com Várias Comodidades (*Linear Multicommodity Min-Cost Flow Problem*) com base no problema dual da relaxação lagrangeana.

Esta tese está organizada da seguinte forma. No Capítulo 2 descreve-se o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto e apresentam-se duas formulações de fluxo orientadas para este problema. A primeira obtém-se de uma conhecida formulação de fluxos para o problema da Árvore de Suporte de Custo Mínimo e a segunda é uma formulação mais compacta e foi obtida por Gouveia [11], utilizando a técnica de redefinição de variáveis de Martin[16]. No Capítulo 3 apresenta-se a técnica dos algoritmos Duais Ascendentes e constrói-se um algoritmo Dual Ascendente para o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto. Começamos por efectuar a descrição da formulação dual linear da formulação alternativa apresentada no capítulo anterior e que serve de base para a descrição do algoritmo Dual Ascendente que construímos e terminamos o capítulo com a apresentação de um exemplo da sua aplicação. No Capítulo 4 apresentamos uma heurística que utiliza a solução dual ascendente como solução de partida e cujo objectivo é o de encontrar um limite superior para o valor óptimo do problema. No Capítulo 5 são apresentados alguns resultados computacionais. Começamos este capítulo com uma breve descrição de como foram geradas várias instâncias do problema e posteriormente apresentamos os resultados que obtivemos com a aplicação do algoritmo Dual Ascendente e da heurística. Os resultados apresentados correspondem a instâncias do problema com 20, 40 e 60 nodos destino, considerando que o valor para o parâmetro H é de 3, 4 e 5. Por fim, no Capítulo 6 apresentamos algumas considerações finais.

Capítulo 2

O Problema HMST

Neste capítulo começamos por definir formalmente o problema e apresentamos duas formulações propostas por Gouveia [10, 11]. Uma formulação de fluxos orientada (MCF) e uma formulação de fluxos orientada com índices de salto (HMCF).

2.1 Descrição do problema

O Problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto, HMST (***H**op-constrained **M**inimum **S**panning **T**ree*), é definido da seguinte forma:

> Seja $\mathcal{G} = (\mathcal{N}, E)$ um grafo simples[1], onde $\mathcal{N} = \{0, 1, ..., n\}$ representa o conjunto dos nodos e $E = \{\{i, j\} : i, j \in \mathcal{N}, i \neq j\}$ o conjunto das arestas, tal que a cada aresta $\{i, j\} \in E$ está associado um custo c_{ij}. Seja ainda, H um número natural.
>
> Dado um nodo raíz (que sem perda de generalidade assumimos ser o nodo 0) o Problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto consiste em determinar a árvore de suporte, T, de custo mínimo[2] com a restrição adicional de que o número de saltos (arestas) no único caminho desde o nodo

[1] isto é, um grafo sem arestas paralelas e lacetes

[2] sendo o custo de T dado por $c(T) = \sum_{\{i,j\} \in T} c_{ij}$

raíz até qualquer outro nodo da árvore não é superior a H.

Considere-se um grafo para o qual se pretende obter a árvore de suporte de custo mínimo com a restrição adicional de que o número de arestas no único caminho entre o nodo raíz (nodo 0) e os restantes nodos do grafo não é superior a 3, ou seja, com $H = 3$. Admita-se que se aplicou o algoritmo de Prim e se obteve a árvore de suporte de custo mínimo que se apresenta na Figura 2.1.

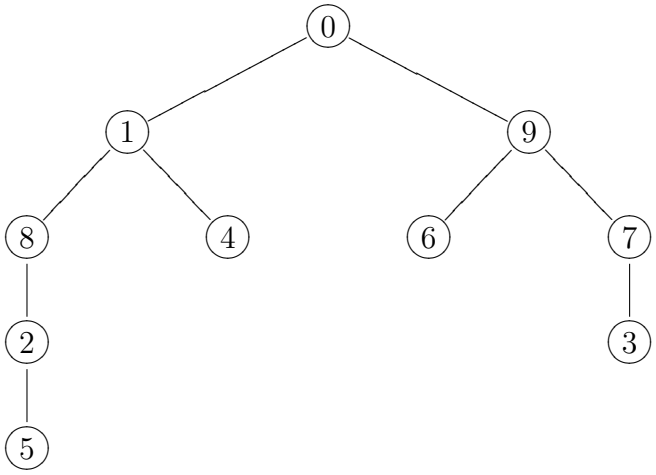


Figura 2.1: Árvore de Suporte de Custo Mínimo

Note que [0,1,8,2,5] é um caminho com 4 arestas, pelo que esta solução não é admissível para o nosso problema pois viola as restrições de salto. Uma solução admissível seria, por exemplo, a árvore da Figura 2.2.

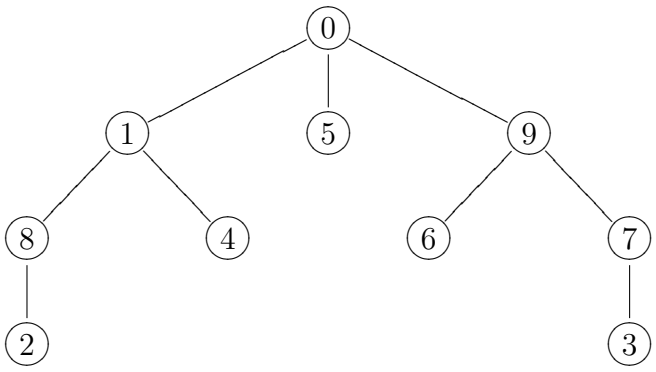


Figura 2.2: Árvore de Suporte com restrição de salto com $H = 3$

Recentes avanços em Optimização Combinatória, indicam que uma melhor formulação (mais compacta e/ou com melhor valor da relaxação linear), para vários problemas de redes, pode ser obtida, definindo o problema num grafo orientado (veja-se, por exemplo, Magnanti e Wolsey[14]). Além disso, formulações orientadas, isto é, definidas em grafos orientados, têm-se mostrado mais adequadas para utilizar em processos de obtenção de limites inferiores[10].

Por esta razão, neste trabalho, apenas consideraremos formulações orientadas para o problema HMST.

Um problema definido num grafo não orientado $\mathcal{G} = (\mathcal{N}, E)$ pode facilmente ser transformado num problema equivalente num grafo orientado, substituindo cada aresta $e = \{i, j\} \in E$, por dois arcos, o arco (i, j) e o arco (j, i), associando a cada um destes novos arcos o custo da aresta que lhes deu origem. Assim, o grafo resultante é orientado e simétrico, isto é $c_{ij} = c_{ji}$, $\forall i, j \in \mathcal{N}$.

O problema de encontrar a árvore de suporte de custo mínimo é então equivalente ao problema de encontrar a arborescência de custo mínimo, com raíz num qualquer nodo, no grafo orientado equivalente. Para simplificar, assumimos que o nodo raíz é o nodo 0. Também assumimos que os arcos da arborescência são dirigidos para fora do nodo raíz, consequentemente, cada aresta $\{0, j\} \in E$ é apenas substituída por um único arco, o arco $(0, j)$.

No que se segue designaremos por $\mathcal{D} = \mathcal{N}\backslash\{0\}$ o conjunto de todos os nodos de $\mathcal{G}$ excepto o nodo raíz, por $\mathcal{A}$ o conjunto dos arcos no grafo orientado, isto é, $\mathcal{A} = \{(i,j) : i \in \mathcal{N}, j \in \mathcal{D}, i \neq j\}$, por $I(j) = \{i : i \in \mathcal{N}\ e\ (i,j) \in \mathcal{A}\}\ (j \in \mathcal{D})$ o conjunto dos nodos adjacentes a j através de arcos convergentes em j e por $J(i) = \{j : j \in \mathcal{D}\ e\ (i,j) \in \mathcal{A}\}\ (i \in \mathcal{N})$ o conjunto dos nodos adjacentes a i através de arcos divergentes de i.

2.2 Formulação de fluxos orientada

Começamos por apresentar uma formulação de fluxos orientada para o problema HMST que designamos por MCF (*Multicommodity Flow Formulation*), apresentada por Gouveia[10]. Esta formulação obtém-se adicionando as restrições de salto a uma conhecida formulação de fluxos para o problema da árvore de suporte de custo mínimo (veja-se, por exemplo, Magnanti e Wolsey[14]).

Consideramos as variáveis binárias orientadas $x_{ij}((i,j) \in \mathcal{A})$ que indicam se o arco (i,j) pertence ou não à árvore de suporte e as variáveis de fluxo orientadas $y_{ij}^k((i,j) \in \mathcal{A},\ k \in \mathcal{D},\ i \neq k)$ que indicam se o arco (i,j) é ou não utilizado no caminho da raíz para o nodo k, ou seja:

$$x_{ij} = \begin{cases} 1 & \text{, se o arco } (i,j) \text{ pertence à árvore de suporte} \\ 0 & \text{, caso contrário} \end{cases}$$

$$y_{ij}^k = \begin{cases} 1 & \text{, se o arco } (i,j) \text{ é utilizado no caminho da raíz até ao nodo k} \\ 0 & \text{, caso contrário} \end{cases}$$

A formulação do problema HMST como um problema de fluxos num grafo orientado tem a seguinte estrutura.

Formulação (MCF):

$$min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}$$

$s.a$

$$\sum_{i\in\mathcal{N}\setminus\{j\}} x_{ij} = 1 \qquad j \in \mathcal{D} \qquad \text{(mcf1)}$$

$$\sum_{i\in\mathcal{N}\setminus\{j,k\}} y_{ij}^k - \sum_{i\in\mathcal{D}\setminus\{j\}} y_{ji}^k = \begin{cases} -1 & j = 0 \\ 0 & j \neq 0, k \\ 1 & j = k \end{cases} \qquad j,k \in \mathcal{D} \qquad \text{(mcf2)}$$

$$y_{ij}^k \leq x_{ij} \qquad (i,j) \in \mathcal{A}, \quad k \in \mathcal{D} \qquad \text{(mcf3)}$$

$$\sum_{(i,j)\in\mathcal{A}} y_{ij}^k \leq H \qquad k \in \mathcal{D} \qquad \text{(mcf4)}$$

$$y_{ij}^k \in \{0,1\} \qquad (i,j) \in \mathcal{A}, \quad k \in \mathcal{D} \qquad \text{(mcf5)}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in \mathcal{A} \qquad \text{(mcf6)}$$

Para simplificar a notação e a escrita da formulação observamos que embora incluidas na formulação, assumimos que as variáveis y_{i0}^k $(i,k \in \mathcal{D})$ e y_{kj}^k $(j,k \in \mathcal{D})$ têm sempre valor nulo.

As restrições (mcf1) garantem que um e apenas um arco chega a cada nodo (excepto para o nodo raíz). As restrições (mcf2) são as usuais restrições de conservação de fluxo. Para cada nodo destino k, garantem que a raíz envia uma unidade de fluxo para k, garantem que o nodo destino k recebe uma unidade de fluxo e para os restantes nodos garantem que a quantidade de fluxo que entra é igual à quantidade de fluxo que sai desse nodo. As restrições (mcf3) são as restrições de ligação que estabelecem a relação entre as variáveis topológicas x_{ij} e as variáveis de fluxo y_{ij}^k. Além disso, garantem que apenas é possível enviar fluxo através do arco (i,j) se este arco se encontra na solução. As restrições (mcf4) são as restrições de salto, e estabelecem que, para cada nodo destino k, não mais do que H arcos são incluídos no caminho entre o nodo raíz e o nodo k. Por fim, as restrições (mcf5) e (mf6) são as restrições de integralidade das variáveis.

Obtemos a relaxação linear da formulação MCF, que designamos por MCF_L, substituindo as restrições (mcf5) e (mcf6), respectivamente, por:

$$y_{ijk} \geq 0 \qquad (i,j) \in \mathcal{A}, \quad k \in \mathcal{D}, \quad i \neq k \qquad \text{(mcf5´)}$$

$$x_{ij} \geq 0 \qquad (i,j) \in \mathcal{A} \qquad \text{(mcf6´)}$$

Note que não é necessário incluir as restrições $y_{ijk} \leq 1$ $((i,j) \in \mathcal{A},\ k \in \mathcal{D},\ i \neq k)$ nem $x_{ij} \leq 1$ $((i,j) \in \mathcal{A})$ pois estão implícitas por (mcf1) e (mcf3).

No seu conjunto as restrições (mcf2), (mcf3), (mcf4) e (mcf5) garantem que a solução determinada é conexa e contém um caminho, com não mais do que H arcos, entre o nodo raíz e cada nodo $k \in \mathcal{D}$. Adicionando a este conjunto as restrições (mcf1) e (mcf6) garantimos que a solução é uma árvore orientada com raíz no nodo 0.

Referimos ainda que (ver Gouveia[10]) a relaxação linear da formulação MCF sem as restrições de salto (mcf4) fornece uma descrição completa do envolvente convexo da árvore orientada de suporte de custo mínimo. O que significa que esta formulação sem as restrições de salto é exacta (no sentido de que a solução óptima da sua relaxação linear é sempre inteira) para o problema da árvore de suporte de custo mínimo.

2.3 Formulação de fluxos orientada com índices de salto

Nesta secção vamos apresentar uma formulação alternativa (mais compacta) para o problema HMST. Esta formulação foi obtida por Gouveia[11] seguindo a técnica de redefinição de variáveis de Martin[16]. A nova formulação, que será designada por HMCF (*Hop-indexed Multicommodity Flow Formulation*), utiliza variáveis de fluxo com índices de salto, e pode ser vista como uma formulação de fluxos com índices de salto.

A nova formulação para o HMST utiliza as variáveis binárias orientadas $x_{ij}((i,j) \in \mathcal{A})$

que indicam se o arco (i,j) pertence ou não à árvore de suporte e as variáveis de fluxo orientadas $z_{ij}^{hk}((i,j) \in \mathcal{A},\ k \in \mathcal{D},\ i \neq k,\ h = 1,..,H)$ que indicam se o arco (i,j) é ou não incluido exactamente na posição h no único caminho da raíz para o nodo k, ou seja:

$$x_{ij} = \begin{cases} 1 & \text{, se o arco } (i,j) \text{ pertence à árvore de suporte de custo mínimo} \\ 0 & \text{, caso contrário} \end{cases}$$

$$z_{ij}^{hk} = \begin{cases} 1 & \text{, se o arco } (i,j) \text{ é incluído exactamente na posição } h \text{ no único caminho da} \\ & \text{raíz para o nodo } k \\ 0 & \text{, caso contrário} \end{cases}$$

A nova formulação para o problema HMST tem a seguinte forma.

Formulação (HMCF):

$$min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij}$$

$$s.a$$

$$\sum_{i\in I(j)} x_{ij} = 1 \qquad j \in \mathcal{D} \qquad \text{(hmcf1)}$$

$$z_{0i}^{1k} - \sum_{j\in J(i)} z_{ij}^{2k} = 0 \qquad i \in J(0),\quad k \in \mathcal{D} \qquad \text{(hmcf2)}$$

$$\sum_{j\in I(i)} z_{ji}^{hk} - \sum_{j\in J(i)\setminus\{k\}} z_{ij}^{h+1,k} = 0 \qquad i,k \in \mathcal{D},\quad h = 2,..,H-1 \qquad \text{(hmcf3)}$$

$$\sum_{j\in I(k)\cup\{k\}} z_{jk}^{Hk} = 1 \qquad k \in \mathcal{D} \qquad \text{(hmcf4)}$$

$$\sum_{h=1}^{H} z_{ij}^{hk} \leq x_{ij} \qquad (i,j) \in \mathcal{A},\quad k \in \mathcal{D} \qquad \text{(hmcf5)}$$

$$z_{0j}^{1k} \in \{0,1\} \qquad j \in J(0),\quad k \in \mathcal{D} \qquad \text{(hmcf6)}$$

$$z_{ij}^{hk} \in \{0,1\} \qquad (i,j) \in \mathcal{A},\quad k \in \mathcal{D},\quad i \neq 0,\quad h = 2,..,H \qquad \text{(hmcf7)}$$

$$z_{kk}^{hk} \in \{0,1\} \qquad k \in \mathcal{D}, \quad h = 2,..,H \tag{hmcf8}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in \mathcal{A} \tag{hmcf9}$$

Para simplificar a notação e a escrita da formulação observamos que embora incluidas na formulação, assumimos que as variáveis $z_{kj}^{hk},\ (j,k \in \mathcal{D},\ h = 1,..,H$ têm sempre valor nulo.

As restrições (hmcf1) garantem que um e apenas um arco chega a cada nodo (excepto para o nodo raíz). Para cada nodo $k \in \mathcal{D}$, as restrições (hmcf2) estabelecem que o arco $(0,i)$ está no caminho para o nodo k na posição 1 se e só se existe um arco deixando o nodo i no caminho para o nodo k na posição 2. As restrições (hmcf3) garantem que, no caminho para o nodo k, um arco entra no nodo i na posição h se e só se um arco deixa o nodo i na posição $h+1$. As restrições (hmcf4) estabelecem que, no caminho para o nodo k, um e apenas um arco entra no nodo k na posição H. As restrições (hmcf5) são as restrições de ligação que estabelecem a relação entre as variáveis x_{ij} e z_{ij}^{hk} e garantem, para cada k, que o arco (i,j) é incluído numa posição, e apenas numa, do caminho mais curto entre o nodo raíz e o nodo k se esse arco é incluído na solução. As restrições (hmcf6), (hmcf7), (hmcf8) e (hmcf9) são as restrições de integralidade das variáveis z_{ij}^{hk} e x_{ij}.

Para definir a relaxação linear deste modelo, que designaremos por $\mathrm{HMCF_L}$, substituímos as restrições de integralidade (hmcf6), (hmcf7), (hmcf8) e (hmcf9), respectivamente, por:

$$z_{0j}^{1k} \geq 0 \qquad j \in J(0), \quad k \in \mathcal{D} \tag{hmcf6´}$$

$$z_{ij}^{hk} \geq 0 \qquad (i,j) \in \mathcal{A}, \quad k \in \mathcal{D}, \quad i \neq 0, \quad h = 2,..,H \tag{hmcf7´}$$

$$z_{kk}^{hk} \geq 0 \qquad k \in \mathcal{D}, \quad h = 2,..,H \tag{hmcf8´}$$

$$x_{ij} \geq 0 \qquad (i,j) \in \mathcal{A} \tag{hmcf9´}$$

Note que não é necessário incluir as restrições $z_{0j}^{1k} \leq 1\ (j \in J(0),\ k \in \mathcal{D})$, $z_{ij}^{hk} \leq 1\ ((i,j) \in \mathcal{A},\ k \in \mathcal{D},\ i \neq 0,\ h = 2,..,H)$, $z_{kk}^{hk} \leq 1\ (k \in \mathcal{D},\ h = 2,..,H)$ nem

$x_{ij} \leq 1$ ($(i,j) \in \mathcal{A}$) pois estão implícitas por (hmcf1) e (hmcf5).

Para cada $k \in \mathcal{D}$ o sistema de restrições (hmcf2), (hmcf3), (hmcf4), (hmcf6), (hmcf7) e (hmcf8) constitui uma formulação compacta e exacta para o problema do caminho mais curto com restrições de salto entre o nodo 0 e o nodo k. Notamos que, para cada k, cada um destes sistemas de equações usam as variáveis $z_{kk}^{hk},\ h = 2, .., H$, e é precisamente por isto que cada um destes sistemas permitem caminhos entre o nodo raíz e o nodo k com menos de H saltos. Quando um caminho contém menos de H arcos são necessários lacetes no nodo k para obter uma solução contendo exactamente H variáveis com valor igual a 1.

Notamos ainda que as variáveis de fluxo y_{ij}^k e as variáveis de fluxo com índices de salto z_{ij}^{hk} estão relacionadas da seguinte forma:

$$\begin{aligned} z_{0j}^{1k} &= y_{0j}^k && (0,j) \in \mathcal{A},\quad k \in \mathcal{D} \\ \sum_{h=2}^{H} z_{ij}^{hk} &= y_{ij}^k && (i,j) \in \mathcal{A},\quad k \in \mathcal{D},\quad i \neq 0 \end{aligned}$$

A formulação dual linear da formulação de fluxo com índices de salto será apresentada no Capítulo 3 e serve de base para a descrição do Algoritmo Dual Ascendente que construímos. Estas formulações serão também utilizadas no Capítulo 5 para determinar o valor óptimo (ou um seu limite, superior ou inferior) das instâncias de teste.

Capítulo 3

Um Algoritmo Dual Ascendente

O objectivo da técnica Dual Ascendente é o de gerar rapidamente uma solução admissível do problema dual ou da relaxação linear ou da relaxação lagrangeana, obtendo-se desta forma um limite para o valor óptimo do problema.

A abordagem Dual Ascendente que utilizámos considera o problema dual da relaxação linear do problema original. Começamos com uma solução dual admissível e iterativamente modificamos os valores das variáveis duais de modo a que, em cada iteração, o valor da função objectivo dual aumente.

Neste capítulo começamos por apresentar o problema dual e a técnica dos algoritmos Duais Ascendentes, posteriormente constrói-se um algoritmo Dual Ascendente para o problema HMST e finalizamos com um exemplo.

3.1 O problema dual e a técnica Dual Ascendente

Consideramos a formulação HMCF_L apresentada no Capítulo 2. Uma vez que assumimos que os custos c_{ij} são não negativos esta formulação mantém-se válida se não incluirmos as restrições (hmcf1). Tal como já referimos o conjunto de restrições (hmcf2), (hmcf3), (hmcf4), (hmcf6), (hmcf8) e (hmcf9) definem os caminhos mais curtos entre o nodo raíz e cada nodo k, com no máximo H arcos e garantem, por isso, a conexidade do grafo. Como

o problema é de minimização e os custos c_{ij} são não negativos, a solução óptima inclui o menor número de arestas possíveis, logo o resultado é uma árvore de suporte.

No que se segue consideramos a formulação HMCF_L sem essas restrições.

Para determinarmos o problema dual linear associamos as variáveis duais γ_i^{1k} $(i \in J(0),\ k \in \mathcal{D})$, γ_i^{hk} $(i,k \in \mathcal{D},\ h = 2,..,H-1)$ e γ_k^{Hk} $(k \in \mathcal{D})$ às restrições (hmcf2), (hmcf3) e (hmcf4), respectivamente. O problema dual tem a seguinte estrutura.

Formulação (DualHMCF_L):

$$max \quad \sum_{k\in\mathcal{D}} \gamma_k^{Hk}$$

$s.a$

$$\gamma_j^{1k} \leq \lambda_{0j}^k; \qquad j,k \in \mathcal{D} \qquad \text{(dual1)}$$

$$\gamma_j^{hk} - \gamma_i^{h-1,k} \leq \lambda_{ij}^k; \qquad (i,j) \in \mathcal{A},\ i \neq 0,\ i \neq k,\ k \in \mathcal{D},\ h = 2,...,H-1 \qquad \text{(dual2)}$$

$$\gamma_k^{Hk} - \gamma_i^{H-1,k} \leq \lambda_{ik}^k; \qquad i,k \in \mathcal{D},\ i \neq k \qquad \text{(dual3)}$$

$$\sum_{k\in\mathcal{D}} \lambda_{ij}^k \leq c_{ij}; \qquad (i,j) \in \mathcal{A} \qquad \text{(dual4)}$$

$$\lambda_{ij}^k \geq 0; \qquad (i,j) \in \mathcal{A},\ k \in \mathcal{D} \qquad \text{(dual5)}$$

As variáveis γ_i^{hk} $(i,k \in \mathcal{D},\ h = 1,..,H)$ não tem qualquer restrição de sinal.

Observamos que para valores fixos das variáveis λ_{ij}^k, $((i,j) \in \mathcal{A},\ k \in \mathcal{D})$ que satisfaçam as restrições (dual4) e (dual5), este problema pode decompor-se em n subproblemas independentes, um para cada $k \in \mathcal{D}$.

O subproblema correspondente ao nodo destino k é:

$$max \quad \gamma_k^{Hk}$$

$s.a$

$$\gamma_j^{1k} \leq \lambda_{0j}^k; \qquad j \in \mathcal{D} \qquad \text{(kdual1)}$$

$$\gamma_j^{hk} - \gamma_i^{h-1,k} \leq \lambda_{ij}^k; \qquad (i,j) \in \mathcal{A}, i \neq 0, h = 2,...,H-1 \qquad \text{(kdual2)}$$

$$\gamma_k^{Hk} - \gamma_i^{H-1,k} \leq \lambda_{ik}^k; \qquad i \in \mathcal{D} \qquad \text{(kdual3)}$$

Se interpretarmos os valores das variáveis λ_{ij}^k como sendo os custos associados a cada arco $(i,j) \in \mathcal{A}$, então cada um destes subproblemas corresponde ao problema dual do seguinte problema do caminho mais curto com restrições de salto entre a origem e o nodo k.

$$
\begin{aligned}
& min \sum_{(i,j)\in\mathcal{A}} \sum_{h=1}^{H} \lambda_{ij}^k z_{ij}^{hk} \\
& s.a \\
& z_{0i}^{1k} - \sum_{j\in J(i)} z_{ij}^{2k} = 0 && i \in J(0), \quad i \neq k \\
& \sum_{j\in I(i)} z_{ji}^{hk} - \sum_{j\in J(i)\setminus\{k\}} z_{ij}^{h+1,k} = 0 && i \in \mathcal{D}, \quad i \neq k, \quad h = 2,..,H-1 \\
& \sum_{j\in I(k)\cup\{k\}} z_{jk}^{Hk} = 1 \\
& z_{0j}^{1k} \geq 0 && j \in J(0) \\
& z_{ij}^{hk} \geq 0 && (i,j) \in \mathcal{A}, \quad i \neq 0,k, \quad h = 2,..,H \\
& z_{kk}^{hk} \geq 0 && h = 2,..,H
\end{aligned}
$$

Portanto, uma vez fixos valores admissíveis para as variáveis $\lambda_{ij}^k((i,j) \in \mathcal{A},\ k \in \mathcal{D})$ facilmente se calcula o caminho mais curto para cada subproblema do caminho mais curto com restrições de salto entre a raíz e o nodo k. Desta forma para aqueles valores de λ_{ij}^k, obtemos valores admissíveis para as variáveis γ_j^{hk}, isto é, obtemos uma solução dual admissível.

Como já referimos a estratégia dos algoritmos duais ascendentes consiste em, partindo de uma solução dual admissível, iterativamente modificar os valores das variáveis duais de modo a aumentar o valor da função objectivo dual. Após termos obtido valores admissíveis para as variáveis γ_j^{hk} interessa ver como podemos modificar o valor das variáveis duais λ_{ij}^k, em cada iteração, de modo a que as restrições duais permaneçam válidas e o valor da

função objectivo dual aumente.

Já vimos que para valores fixos das variáveis λ_{ij}^k, o problema pode separar-se em vários subproblemas, e portanto, se atribuirmos valores admissíveis para as variáveis λ_{ij}^k facilmente se calculam os caminhos mais curtos para cada subproblema, obtendo-se desta forma valores admissíveis para γ_i^{hk} e, consequentemente, uma solução dual admissível.

Assumindo que γ_j^{hk} representa o valor do caminho mais curto, com no máximo h arcos, entre a raíz e o nodo j no caminho da raíz para o nodo k que passa no nodo j, o valor óptimo deste subproblema, γ_k^{Hk}, representa o valor do caminho mais curto entre o nodo raíz e o nodo k com no máximo H arcos. Portanto, para aumentarmos o valor da função objectivo dual, temos de aumentar os valores dos caminhos mais curtos para um ou mais nodos destinos. Mas, para que isso aconteça é necessário aumentar o valor das variáveis λ_{ij}^k dos arcos que se encontram em caminhos para os nodos $k \in \mathcal{D}$, mantendo a admissibilidade dual.

A estratégia do método Dual Ascendente pode ser resumida da seguinte forma.

Iterativamente, aumentar o valor de uma ou mais variáveis duais λ_{ij}^k $((i,j) \in \mathcal{A},\ k \in \mathcal{D})$ de modo a que:

1. a solução se mantenha dual admissível, em particular $\sum_{k \in \mathcal{D}} \lambda_{ij}^k \leq c_{ij}$ com $\lambda_{ij}^k \geq 0$, $\forall k \in \mathcal{D}$
2. o comprimento do caminho mais curto com no máximo H arcos, γ_k^{Hk}, aumente para pelo menos um nodo destino k em cada iteração.

Assim, para que as condições anteriores sejam satisfeitas temos de definir quais as variáveis λ_{ij}^k que podem aumentar de valor e qual o aumento que é possível efectuar, garantindo que os valores actualizados satisfazem as restrições (dual4) e (dual5) e que provocam obrigatoriamente um aumento em γ_k^{Hk} para pelo menos algum k. Deste modo, para satisfazer a condição 1, as variáveis λ_{ij}^k $((i,j) \in \mathcal{A},\ k \in \mathcal{D})$ que podem aumentar de valor são apenas aquelas para as quais as variáveis de folga $s_{ij} = c_{ij} - \sum_{k \in \mathcal{D}} \lambda_{ij}^k$ tem valor positivo. Para satisfazer a condição 2, temos de identificar os arcos (i,j) que se encon-

tram em caminhos para o nodo k e cujos respectivos valores das variáveis λ_{ij}^k possam ser aumentados, de modo a que o custo de um caminho admissível para um nodo k possa aumentar. Suponha-se que, em determinada iteração do algoritmo, temos uma solução dual admissível e que os valores das variàveis λ_{ij}^k satisfazem a correspondente restrição (dual4) para um arco (i, j) de modo estrito, o que significa que o custo c_{ij} desse arco não é totalmente usado, ou seja, $s_{ij} > 0$. A estratégia dual ascendente consiste em, usando as variáveis s_{ij}, aumentar o valor das variáveis λ_{ij}^k para um ou mais nodos destino $k \in \mathcal{D}$ de forma a que o comprimento γ_k^{Hk} do caminho mais curto entre a raíz e esses nodos k aumente. Essencialmente, o algoritmo faz uma procura dos custos c_{ij} ainda não usados para, selectivamente, fazer uso desses custos e distribuí-los de forma a aumentar o comprimento do caminho mais curto com restrições de salto entre a origem e um ou mais nodos k.

3.2 Um algoritmo Dual Ascendente para o problema HMST

Uma vez definida a estratégia geral vamos ver como é a que podemos aplicar ao nosso problema.

O algoritmo Dual Ascendente que propomos obtém iterativamente uma rede de caminhos mais curtos com restrições de salto. Inicializamos com a rede nula, e em cada iteração do algoritmo apenas um nodo destino é considerado e um arco adicionado à rede. Os caminhos são determinados de trás para a frente, isto é, começamos com um determinado nodo e vamos construindo o caminho (com não mais do que H arcos) até ao nodo raíz, o que significa que vamos adicionando arcos à rede até que se atinja o nodo raíz. Uma vez encontrado um caminho mais curto, com não mais do que H arcos, entre esse nodo e o nodo raíz, repetimos este processo para um outro nodo destino. Terminamos quando este procedimento tiver sido realizado para todos os nodos destino.

Vejamos então, como é que, em cada iteração do algoritmo, selecionamos o arco a incluir na solução.

Suponha-se que, em determinada iteração do algoritmo, temos uma solução dual admissível e que estamos a determinar um caminho mais curto com não mais do que H arcos para o nodo k e admita-se que, ainda não se determinou tal caminho.

Para cada nodo destino $k \in \mathcal{D}$ representamos por $Inc(k)$ o conjunto dos nodos que pertencem a pelo menos um caminho para o nodo k com não mais do que $H-1$ arcos e por $Exc(k)$ o conjunto definido da seguinte forma: $Exc(k) = \mathcal{N} \backslash Inc(k)$, isto é, inclui os nodos que não pertencem a qualquer caminho para o nodo k. Note que se assumirmos que já existiam caminhos com H arcos, então não seria possível expandir esses caminhos em direcção à raíz por forma a obter um caminho da raíz para o nodo k com no máximo H arcos.

Como exemplo, consideremos um grafo completo cujo conjunto dos nodos é $\mathcal{N} = \{0, 1, 2, 3, 4, 5\}$, onde o nodo 0 representa o nodo raíz e admitimos que $H = 3$. Suponhamos que $k = 5$ e que já determinámos os caminhos $[1, 2, 5]$ e $[3, 5]$. Os nodos $1, 2$ e 3 pertencem a caminhos para o nodo 5 com não mais do que 2 saltos logo temos $Inc(5) = \{1, 2, 3, 5\}$ e $Exc(5) = \{0, 4\}$.

Observamos que $(Exc(k), Inc(k))$ define uma partição do conjunto dos nodos do grafo[1], e dado que assuminos que ainda não existe nenhum caminho entre a raíz e o nodo k, tem-se que o nodo raíz pertence ao conjunto $Exc(k)$ e o nodo k ao conjunto $Inc(k)$.

Representamos por $\mathcal{A}(k)$ o corte induzido pela partição $(Exc(k), Inc(k))$, isto é, $\mathcal{A}(k) = \{(i, j) \in \mathcal{A} : i \in Exc(k),\ j \in Inc(k)\}$, e seja $\mathcal{A}_H(k)$ o subconjunto de arcos de $\mathcal{A}(k)$ tais que ao adicionarmos um arco desse conjunto, ao caminho desde a raíz ao nodo k, originaria caminhos com mais do que H arcos, violando, portanto, as restrições de salto.

Em relação ao exemplo anterior, temos $\mathcal{A}(5) = \{(0, 1), (0, 2), (0, 3), (0, 5), (4, 1), (4, 2), (4, 3), (4, 5)$. No entanto, o arco $(4, 1)$ não pode ser utilizado para expandir um caminho para o nodo 5, pois e o fizessemos obteríamos o caminho $[4, 1, 2, 5]$ que já contém 3 arcos e portanto nenhum caminho válido para o nodo 5 podia ser obtido utilizando este caminho. Assim, $\mathcal{A}_3(5) = \{(4, 1)\}$.

[1]Diz-se que (P_1, P_2) define uma partição do conjunto P se $P_1 \cap P_2 = \emptyset$ e $P_1 \cup P_2 = P$.

Estes conjuntos ajudam na identificação dos arcos (i, j) tais que um aumento no valor das correspondentes variáveis λ_{ij}^k podem conduzir a um aumento no custo do caminho mais curto da raíz para o nodo k.

O conjunto $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$ define um corte que separa a raíz do nodo k e define assim, o conjunto dos arcos candidatos a incluir na solução. Ao aumentarmos os valores de λ_{ij}^k para todos os arcos (i, j) neste corte, garantimos que o comprimento γ_k^{Hk} do caminho mais curto entre o nodo raíz e o nodo k, com não mais do que H arcos, aumenta de valor, uma vez que cada um destes arcos pertence a pelo menos a um caminho para o nodo k com não mais do que H arcos.

Uma vez definidos quais os arcos possíveis de incluir na solução, ou equivalentemente, quais os valores de λ_{ij}^k a actualizar, falta definir qual o aumento que se pode efectuar no valor destas variáveis por forma a garantir a admissibilidade da solução dual.

Relembremos que os λ_{ij}^k têm de ser tais que:

$$\sum_{k\in\mathcal{D}} \lambda_{ij}^k \leq c_{ij}; \qquad (i,j) \in \mathcal{A} \tag{dual4}$$

$$\lambda_{ij}^k \geq 0; \qquad (i,j) \in \mathcal{A}, k \in \mathcal{D} \tag{dual5}$$

E que as variáveis de folga s_{ij} associadas às restrições (dual4), são

$$s_{ij} = c_{ij} - \sum_{k\in\mathcal{D}} \lambda_{ij}^k, \forall (i,j) \in \mathcal{A}.$$

Para cada arco (i, j) de $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$, os valores das variáveis s_{ij} permitem determinar o aumento máximo que se pode efectuar em cada λ_{ij}^k de modo a que as restrições (dual4) e (dual5) permanençam válidas, assim,

$$\delta = min\left\{s_{ij} : (i,j) \in \mathcal{A}(k)\backslash\mathcal{A}_H(k)\right\}$$

define o maior aumento que se pode efectuar nos valores das variáveis λ_{ij}^k, garantindo a admissibilidade da solução dual.

Quando todos os valores das variáveis λ_{ij}^k correspondentes a arcos do corte $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$ aumentam de uma quantidade δ, há pelo menos uma variável s_{ij} que atinge o valor zero,

o que significa que o custo fixo c_{ij} desse arco é atingido e que houve expansão de um caminho para o nodo k em mais um arco, portanto, o arco (i, j) passa a fazer parte da solução.

Acabámos de definir uma iteração do algoritmo Dual Ascendente que construímos. Em seguida, fazemos uma descrição completa desse algoritmo, que automatiza todos estes procedimentos que permitem encontrar pelo menos um caminho com restrição de salto entre a raíz e cada nodo k.

Tal como tinhamos referido o algoritmo parte de uma solução dual admissível. No algoritmo que construímos a solução inicial é a solução nula, que é, como facilmente se verifica, uma solução dual admissível. Por este motivo todas as variáveis duais λ_{ij}^k e γ_j^{hk} são inicializadas a zero, e como inicialmente todos os arcos têm custo nulo, então as variáveis s_{ij} tomam valor inicial igual a c_{ij}.

Utilizaremos o conjunto $Dest$ para representar o conjunto dos nodos destino para os quais ainda não foi determinado um caminho para a origem e a variável *custo* que indica o valor da função objectivo dual. Inicialmente o conjunto $Dest$ coincide com o conjunto $\mathcal{D}$ e o valor da função objectivo dual corresponde ao valor da solução inicial que é 0, ou seja, $custo = 0$.

Em cada iteração do algoritmo seleccionamos um nodo k pertencente ao conjunto $Dest$ e inicializamos os conjuntos $Inc(k)$ e $Exc(k)$. Determinamos da forma atrás descrita o arco a incluir na solução, que designaremos por (i_{aux}, j_{aux}), o que significa que tentamos expandir um caminho para esse nodo destino que não viole as restrições de salto. Posteriormente actualizamos os valores das variáveis duais e os conjuntos $Inc(k)$ e $Exc(k)$. Repetimos o processo (isto é vamos introduzindo arcos) até que a raíz seja alcançada, isto é, até que o arco introduzido seja da forma $(0, j_{aux})$. Quando $i_{aux} = 0$ isso significa que foi encontrado um caminho com não mais do que H arcos entre o nodo raíz e o nodo k. Nessa altura o nodo k deixa de pertencer ao conjunto dos nodos $Dest$ e o algoritmo recomeça um novo ciclo para um outro nodo k. Quando o conjunto $Dest = \emptyset$ isso significa que foi encontrado pelo menos um caminho com restrições de salto entre a raíz e cada nodo $k \in \mathcal{D}$ e o algoritmo termina.

Para que possamos identificar o conjunto $\mathcal{A}_H(k)$, definimos, para cada nodo k, as variáveis $Pos_k(j)$ que indicam a posição do nodo j no caminho da raíz para o nodo k. No início de cada ciclo inicializamos a variável $Pos_k(k) = H$ e sempre que introduzimos um arco actualizamos a variável $Pos_k(i_{aux})$ da seguinte forma $Pos_k(i_{aux}) = Pos_k(j_{aux}) - 1$. Notamos que apenas o nodo raíz pode ter posição igual a 0 e quando um nodo tem posição 1 isso significa que apenas podemos utilizar arcos incidentes na origem para expandir esses caminhos.

O algoritmo pode ser descrito da seguinte forma:

Passo 0. Inicialização

Fazer: $Dest = \mathcal{D}$

$\lambda_{ij}^k = 0, \quad \forall (i,j) \in \mathcal{A}, \quad k \in \mathcal{D}$

$\gamma_i^{hk} = 0, \quad \forall i,k \in \mathcal{D}, \quad h = 1,..,H-1$

$\gamma_k^{Hk} = 0, \quad \forall k \in \mathcal{D}$

$s_{ij} = c_{ij}, \quad \forall (i,j) \in \mathcal{A}$

$custo = 0$

Passo 1. Passo principal

1.1. *Seleccionar um nodo destino, k*

Seleccionar $k \in Dest$

Fazer: $Inc(k) = \{k\}$

$Exc(k) = \mathcal{N} \setminus \{k\}$

$Pos_k(k) = H$

1.2. *Determinar o conjunto dos arcos, $\mathcal{A}(k) \setminus \mathcal{A}_H(k)$, candidatos a incluir na solução*

$\mathcal{A}(k) = \{(i,j) \in \mathcal{A} : i \in Exc(k),\ j \in Inc(k)\}$

$\mathcal{A}_H(k) = \emptyset$

Para todo o $(i,j) \in \mathcal{A}(k)$ fazer:

$$se\ Pos_k(j) = 1 \text{ e } i \neq 0 \ \ então\ \mathcal{A}_H(k) = \mathcal{A}_H(k) \cup \{(i,j)\}$$

1.3. *Determinar o arco, (i_{aux}, j_{aux}), que vamos incluir na solução*

Determinar $\delta = min\{s_{ij} : (i,j) \in \mathcal{A}(k) \backslash \mathcal{A}_H(k)\}$

Seja (i_{aux}, j_{aux}) o arco a que corresponde este valor de δ.

1.4. *Actualização da solução*

Para todo o $(i,j) \in \mathcal{A}(k) \backslash \mathcal{A}_H(k)$ fazer:

$$\lambda_{ij}^k = \lambda_{ij}^k + \delta$$

$$s_{ij} = s_{ij} - \delta$$

Se $\mathcal{A}_H(k) = \emptyset$ *então* fazer:

$$\gamma_j^{hk} = \gamma_j^{hk} + \delta, \text{ para todo o } j \in Inc(k),\ h = 1, .., H-1$$

$$\gamma_k^{Hk} = \gamma_k^{Hk} + \delta$$

senão fazer:

$$\gamma_j^{1k} = \gamma_j^{1k} + \delta, \text{ para todo o } j \in Inc(k)$$

$$\gamma_j^{2k} = \gamma_j^{2k} + \delta, \text{ para todo o } j \in Inc(k) \text{ tal que } Pos(j) \geq 2$$

Se $\mathcal{A}_H(k) = \emptyset$ *então* $custo = custo + \delta$

$Pos_k(i_{aux}) = Pos_k(j_{aux}) - 1$

$Inc(k) = Inc(k) \cup \{i_{aux}\}$

$Exc(k) = Exc(k) \backslash \{i_{aux}\}$

1.5. *Condição de paragem do ciclo*

Se $i_{aux} = 0$ *então* $Dest = Dest \backslash \{k\}$,

ir para o Passo 2.

senão ir para o Passo 1.2.

Passo 2: Critério de Paragem

Se $Dest = \emptyset$ *então* **STOP**.

senão voltar ao Passo 1.

Antes de passarmos à aplicação do algoritmo que acabámos de descrever vejamos algumas observações.

O algoritmo, tal como descrito, apenas fornece o valor da solução dual admissível encontrada e não a solução dual, no entanto, notamos que esta é possível de determinar. No final de cada ciclo identifica-se o caminho determinado para esse nodo e a solução dual ascendente é a junção de todos esses caminhos.

As soluções produzidas pelo algoritmo dual ascendente satisfazem a seguinte propriedade.

Propriedade: Caminho mais Curto

1. Para cada nodo $k \in \mathcal{D}$ todos os nodos $j \in Inc(k)$ pertencem a pelo menos um caminho para o nodo k, com não mais do que H arcos.

2. Em cada iteração do algoritmo, usando os custos modificados λ_{ij}^k, γ_j^{hk} representa o valor do caminho mais curto, com no máximo h arcos, entre a raíz e o nodo j no caminho da raíz para o nodo k que passa no nodo j.

Inicialmente, o conjunto $Inc(k)$ contém apenas o nodo k e, portanto, a condição 1 é satisfeita. Numa qualquer iteração do algoritmo, um nodo i é transferido do conjunto $Exc(k)$ para o conjunto $Inc(k)$ apenas se um arco (i, j) do corte $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$ tiver a correspondente variável s_{ij} com valor nulo e, portanto a inclusão deste arco não viola a restrição de salto. Assim, o nodo i passa a fazer parte de um caminho com restrição de salto para o nodo k.

Relativamente à segunda condição. Inicialmente todos os arcos tem custo nulo, logo qualquer caminho entre a raíz e o nodo k tem valor nulo. Uma vez que todas as variáveis

γ_j^{hk} são inicializados a zero esta condição é satisfeita. Numa qualquer iteração do algoritmo, quando aumentamos o valor das variáveis λ_{ij}^k para os arcos pertencentes ao corte $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$, não alteramos o valor dos caminhos já construídos. Assim, para os novos custos, e uma vez que os custos dos arcos mantêm ou aumentam δ o valor dos caminhos também mantêm ou aumentam δ. Se $\mathcal{A}_H(k) = \emptyset$ então o valor dos caminhos para o nodo k, γ_j^{hk}, aumentam para todo o j que pertença a um caminho para o nodo k, uma vez que esses caminhos incluem obrigatoriamente um arco de $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$, logo neste caso a condição 2 é satisfeita. Vejamos agora que quando $\mathcal{A}_H(k) \neq \emptyset$ a condição 2 também é satisfeita. Se $\mathcal{A}_H(k) \neq \emptyset$ então existe pelo menos um nodo diferente da raíz pertencente ao conjunto $Exc(k)$ e um nodo $m \in Inc(k)$ que se encontra na posição 1 no caminho da raíz para o nodo k que passa no nodo m. Para todo o j que pertença a um caminho para o nodo k, γ_j^{1k} aumenta δ uma vez que estes caminhos são da forma [0,j] e todos os arcos da forma $(0, j)$ pertence ao corte $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$. Para todo o j que pertença a um caminho para o nodo k e que se encontre numa posição diferente de 1, γ_j^{2k} também aumenta δ, uma vez que os caminhos com no máximo 2 arcos são da forma $[0, j]$ ou $[0, l, j]$ onde $l \in (k)$ ou $[0, p, j]$ onde $p \in Inc(k)$ e todos os arcos da forma $(0, j), (l, j)$ e (p, j) pertencem ao corte $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$. Para estes nodos γ_j^{hk} com $h \geq 3$ não aumenta de valor uma vez que existe um caminho da forma $[0, l, m, j]$ onde $l \in (k)$ e $m \in Inc(k)$ e se encontra na posição 1 num caminho para k que não aumenta de valor uma vez que utiliza apenas arcos que não pertencem ao corte $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$. Naturalmente $\gamma_j^{hk}, h = 1, .., H-1$ não aumentam de valor para todo o $j \in Exc(k)$, uma vez que estes caminhos utilizam apenas nodos pertencentes a $Exc(k)$ e portanto os respectivos arcos não pertencem a $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$.

Esta propriedade vem também mostrar que o algoritmo mantem sempre a admissibilidade dual. A actualização das variáveis λ_{ij}^k foi determinada por forma a garantir que, em cada iteração do algoritmo, as restrições (dual4) e (dual5), permanecessem válidas. Uma vez mostrado, que em cada iteração, γ_j^{hk} representa o valor do caminho mais curto, com no máximo h arcos, entre a raíz e o nodo j no caminho da raíz para o nodo k que passa no nodo j, usando os custos modificados λ_{ij}^k, mostramos também que as restrições (dual1), (dual2) e (dual3) são verificadas.

Uma outra observação importante prende-se com o tipo da solução dual ascendente. Em cada iteração, o arco (i, j) introduzido na solução é tal que o correspondente valor de s_{ij} é nulo, o que significa (como já referimos) que o custo c_{ij} desse arco é atingido. Notamos que, sempre que o valor da variável s_{ij} se anula, para um determinado arco (i, j) esse valor mantém-se até ao final. Assim, no final a solução dual ascendente contém apenas arcos cujas respectivas variáveis de desvio são nulas, o que implica que o custos dos arcos pertencentes à solução dual ascendente são os custos originais, ou por outras palavras a solução dual ascendente corresponde a um subgrafo do grafo original.

O algoritmo garante que, para cada nodo destino, se obtém pelo menos um caminho válido, mas não garante que esse caminho seja único nem que, no caso de existir mais do que um caminho para um determinado nodo, todos os caminhos para esse nodo sejam válidos. Isto quer dizer que, o algoritmo não garante que no final se obtenha uma solução primal admissível. Os caminhos para cada nodo destino são determinados em separado, ou seja, quando estamos a determinar o caminho mais curto para um determinado nodo, não entramos em linha de conta com os caminhos já determinados para esse nodo. Como a solução dual ascendente é a junção de todos esses caminhos, isso implica que a solução dual ascendente que obtemos pode conter mais do que um caminho para cada nodo destino, caminhos esses que podem não ser todos válidos, isto é, podemos obter caminhos com mais do que H arcos. Por esse motivo a solução dual ascendente não é necessariamente uma árvore de suporte com restrição de salto e portanto não é obrigatoriamente uma solução primal admissível.

Para ilustrar o que estamos a dizer consideremos, como exemplo, um grafo cujo conjunto dos nodos é $\mathcal{N} = \{0, 1, 2, 3, 4, 5\}$, onde (como habitualmente) o nodo 0 representa o nodo raíz e admitimos que $H = 3$. Suponhamos que já determinámos os caminhos para os nodos 4 e 5, sendo esses caminhos $[0, 1, 2, 4]$ e $[0, 2, 3, 5]$, respectivamente.

Figura 3.1: Caminho determinado para o nodo 4

Facilmente se verifica que ambos os caminhos são válidos, e pela forma como se constrói a solução dual ascendente, sabemos que ambos pertencem à solução dual ascendente. As-

Figura 3.2: Caminho determinado para o nodo 5

sim, o grafo que apresentamos na Figura 3.3 é o subgrafo da solução dual ascendente induzido por estes dois caminhos.

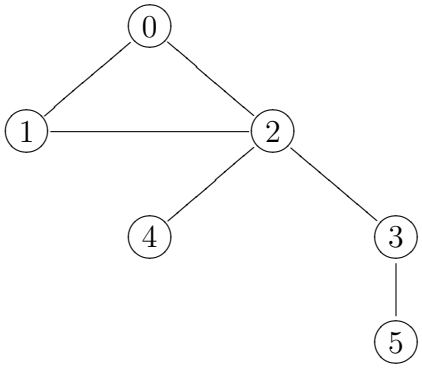


Figura 3.3: Subgrafo da Solução Dual Ascendente

Notamos que apesar de existir para todos os nodos pelo menos um caminho válido, a solução contém pelo menos um ciclo ([0,1,2,0]) e um caminho não válido, o caminho [0,1,2,3,5] contém 4 arestas. Deste modo a solução obtida não se trata de uma árvore de suporte com restrição de salto e por isso não é primal admissível, contudo contém uma solução primal admissível (ver Figura 3.4).

Esta observação sugere que depois de obtermos a solução dual ascendente podemos aplicar algum procedimento que nos permita obter uma solução primal admissível, ou seja, uma árvore de suporte de caminhos válidos. Este assunto será tratado no capítulo seguinte.

Como última observação chamamos a atenção para o facto de que não termos definido nenhuma regra nem para decidir qual o nodo destino a considerar em cada iteração nem qual a escolha do arco a introduzir na solução no caso de haver empate. A escolha que fazemos nestas situações poderá influenciar a solução Dual Ascendente e consequentemente

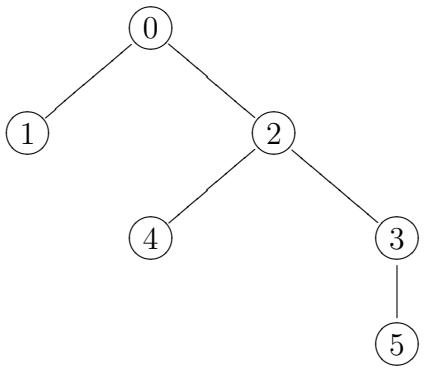


Figura 3.4: Subgrafo da Solução Dual Ascendente correspondente a uma solução primal admissível

a qualidade do limite superior que se obtém.

No exemplo que apresentamos e nos testes computacionais que realizámos considerámos as seguinte regras. Em relação à escolha do nodo a considerar em cada iteração, optámos por começar por determinar o caminho mais curto com restrição de salto para o nodo n, depois calculamos o caminho para o nodo $n-1$ e assim por diante até atingirmos o nodo 1. Em relação ao arco a introduzir na solução em caso de empate optámos por introduzir o arco que aparecia em primeiro lugar no conjunto $\mathcal{A}(k)\backslash\mathcal{A}_H(k)$. Pensamos que poderá ser interessante considerar outras regras e comparar os resultados obtidos.

3.3 Exemplo de Aplicação

Nesta secção vamos exemplificar o algoritmo Dual Ascendente descrito na secção anterior, para tal consideremos um grafo simples completo com 6 nodos, $\mathcal{N} = \{0, 1, 2, 3, 4, 5\}$, e cuja matriz de custos (simétrica), C, é:

$$C = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[\begin{array}{cccccc} - & 5 & 3 & 8 & 5 & 9 \\ 5 & - & 1 & 6 & 3 & 6 \\ 3 & 1 & - & 6 & 6 & 2 \\ 8 & 6 & 6 & - & 5 & 7 \\ 5 & 3 & 6 & 5 & - & 8 \\ 9 & 6 & 2 & 7 & 8 & - \end{array}\right] \end{array}$$

Considere ainda que se definiu que o número máximo de saltos permitido era 3, ou seja, $H = 3$, e que o nodo raíz é (como habitualmente) o nodo 0.

Para que seja mais fácil acompanhar a aplicação do algoritmo os valores das variáveis λ_{ij}^k, γ_j^{hk} e $Pos_k(j)$ serão apresentados na forma de matriz que representaremos, respectivamente, por λ^k, γ^k e Pos_k, para cada k. Assim, $\lambda^k = \left[\lambda_{ij}^k\right]_{i=0,..,5;j=1,..,5}$, $\gamma^k = \left[\gamma_j^{hk}\right]_{j=1,..,5;h=1,..,3}$ e $Pos_k = [Pos_k(j)]_{j=0,..,5}$.
As variáveis de folga (s_{ij}) serão também apresentadas na forma de uma matriz, representada por S. Neste caso tem-se $S = [s_{ij}]_{i=0,..,5;j=1,..,5}$.

Iteração nº1:

0. Inicialização:

$Dest = \mathcal{D} = \{1, 2, 3, 4, 5\}$

$\lambda_{ij}^k = 0, \quad \forall (i,j) \in \mathcal{A}, \quad k \in \mathcal{D}$

$\gamma_i^{hk} = 0, \quad \forall i, k \in \mathcal{D}, \quad h = 1, 2$

$\gamma_k^{3k} = 0, \quad \forall k \in \mathcal{D}$

$s_{ij} = c_{ij}, \quad \forall (i,j) \in \mathcal{A}$

$$\text{assim, } S = \begin{bmatrix} 5 & 3 & 8 & 5 & 9 \\ - & 1 & 6 & 3 & 6 \\ 1 & - & 6 & 6 & 2 \\ 6 & 6 & - & 5 & 7 \\ 3 & 6 & 5 & - & 8 \\ 6 & 2 & 7 & 8 & - \end{bmatrix}$$

$custo = 0$

1.1. Seleccionamos o nodo $k = 5$, ou seja, começamos por determinar o caminho para o nodo 5.

$Inc(5) = \{5\}, \quad Exc(5) = \{0, 1, 2, 3, 4\}$

$Pos_5(5) = 3$

$$\text{logo, } Pos_5 = \begin{bmatrix} - & - & - & - & - & 3 \end{bmatrix}$$

As matrizes λ^5 e γ^5 foram inicializadas no passo anterior da seguinte forma:

$$\lambda^5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ - & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^5 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & 0 \end{bmatrix}$$

1.2. Tem-se que:

$\mathcal{A}(5) = \{(0,5),(1,5),(2,5),(3,5),(4,5)\}$

$\mathcal{A}_3(5) = \emptyset$

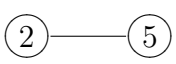
logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(5) \backslash \mathcal{A}_3(5) = \{(0,5),(1,5),(2,5),(3,5),(4,5)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{05}, s_{15}, s_{25}, s_{35}, s_{45},\} = min\{9,6,2,7,8\} = 2 = s_{25}$

logo, vamos incluir o arco $(2,5)$



1.4 As variáveis cujos valores vão ser alterados são:

$\lambda_{05}^5 = \lambda_{15}^5 = \lambda_{25}^5 = \lambda_{35}^5 = \lambda_{45}^5 = 2$

$\gamma_5^{15} = \gamma_5^{25} = \gamma_5^{35} = 2$

$s_{05} = 7, \quad s_{15} = 4, \quad s_{25} = 0, \quad s_{35} = 5, \quad s_{45} = 6$

assim,

$$\lambda^5 = \begin{bmatrix} 0 & 0 & 0 & 0 & \mathbf{2} \\ - & 0 & 0 & 0 & \mathbf{2} \\ 0 & - & 0 & 0 & \mathbf{2} \\ 0 & 0 & - & 0 & \mathbf{2} \\ 0 & 0 & 0 & - & \mathbf{2} \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^5 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ \mathbf{2} & \mathbf{2} & \mathbf{2} \end{bmatrix}; \qquad S = \begin{bmatrix} 5 & 3 & 8 & 5 & \mathbf{7} \\ - & 1 & 6 & 3 & \mathbf{4} \\ 1 & - & 6 & 6 & \mathbf{0} \\ 6 & 6 & - & 5 & \mathbf{5} \\ 3 & 6 & 5 & - & \mathbf{6} \\ 6 & 2 & 7 & 8 & - \end{bmatrix}$$

$custo = 2$

$Inc(5) = \{2, 5\}, \quad Exc(5) = \{0, 1, 3, 4\}$

$Pos_5(2) = 2$

logo, $Pos_5 = \begin{bmatrix} - & - & 2 & - & - & 3 \end{bmatrix}$

1.5. Ir para o passo 1.2.

Iteração nº2:

1.2. Tem-se que:

$\mathcal{A}(5) = \{(0,2), (0,5), (1,2), (1,5), (3,2), (3,5), (4,2), (4,5)\}$

$\mathcal{A}_3(5) = \emptyset$

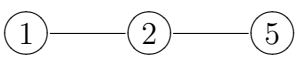
logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(5) \backslash \mathcal{A}_3(5) = \{(0,2), (0,5), (1,2), (1,5), (3,2), (3,5), (4,2), (4,5)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{02}, s_{05}, s_{12}, s_{15}, s_{32}, s_{35}, s_{42}, s_{45}\} = min\{3, 7, 1, 4, 6, 5, 6, 6\} = 1 = s_{12}$

logo, vamos incluir o arco $(1, 2)$



1.4. As variáveis cujos valores vão ser alterados são:

$\lambda_{02}^5 = \lambda_{12}^5 = \lambda_{32}^5 = \lambda_{42}^5 = 1, \quad \lambda_{05}^5 = \lambda_{15}^5 = \lambda_{35}^5 = \lambda_{45}^5 = 3$

$\gamma_2^{15} = \gamma_2^{25} = 1, \quad \gamma_5^{15} = \gamma_5^{25} = \gamma_5^{35} = 3$

$s_{02} = 2, \quad s_{12} = 0, \quad s_{32} = 5, \quad s_{42} = 5, \quad s_{05} = 6, \quad s_{15} = 3, \quad s_{35} = 4, \quad s_{45} = 5$

assim,

$$\lambda^5 = \begin{bmatrix} 0 & \mathbf{1} & 0 & 0 & \mathbf{3} \\ - & \mathbf{1} & 0 & 0 & \mathbf{3} \\ 0 & - & 0 & 0 & 2 \\ 0 & \mathbf{1} & - & 0 & \mathbf{3} \\ 0 & \mathbf{1} & 0 & - & \mathbf{3} \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^5 = \begin{bmatrix} 0 & 0 & - \\ \mathbf{1} & \mathbf{1} & - \\ 0 & 0 & - \\ 0 & 0 & - \\ \mathbf{3} & \mathbf{3} & \mathbf{3} \end{bmatrix}; \qquad S = \begin{bmatrix} 5 & \mathbf{2} & 8 & 5 & \mathbf{6} \\ - & \mathbf{0} & 6 & 3 & \mathbf{3} \\ 1 & - & 6 & 6 & 0 \\ 6 & \mathbf{5} & - & 5 & \mathbf{4} \\ 3 & \mathbf{5} & 5 & - & \mathbf{5} \\ 6 & 2 & 7 & 8 & - \end{bmatrix}$$

$custo = 3$

$Inc(5) = \{1, 2, 5\}, \quad Exc(5) = \{0, 3, 4\}$

$Pos_5(1) = 1$

logo, $Pos_5 = \begin{bmatrix} - & 1 & 2 & - & - & 3 \end{bmatrix}$

1.5. Ir para o passo 1.2.

Iteração nº3:

1.2. Tem-se que:

$\mathcal{A}(5) = \{(0,1), (0,2), (0,5), (3,1), (3,2), (3,5), (4,1), (4,2), (4,5)\}$

$\mathcal{A}_3(5) = \{(3,1), (4,1)\}$

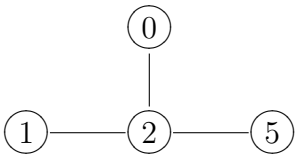
logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(5) \backslash \mathcal{A}_3(5) = \{(0,1), (0,2), (0,5), (3,2), (3,5), (4,2), (4,5)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{01}, s_{02}, s_{05}, s_{32}, s_{35}, s_{42}, s_{45}\} = min\{5, 2, 6, 5, 4, 5, 5\} = 2 = s_{02}$

logo, vamos incluir o arco $(0, 2)$



1.4. As variáveis cujos valores vão ser alterados são:

$\lambda_{01}^5 = 2, \quad \lambda_{02}^5 = \lambda_{32}^5 = \lambda_{42}^5 = 3, \quad \lambda_{05}^5 = \lambda_{35}^5 = \lambda_{45}^5 = 5$

$\gamma_1^{15} = 2, \quad \gamma_2^{15} = \gamma_2^{25} = 3, \quad \gamma_5^{15} = \gamma_5^{25} = 5$

$s_{01} = 3, \quad s_{02} = 0, \quad s_{32} = 3, \quad s_{42} = 3, \quad s_{05} = 4, \quad s_{35} = 2, \quad s_{45} = 3$

assim,

$$\lambda^5 = \begin{bmatrix} \mathbf{2} & \mathbf{3} & 0 & 0 & \mathbf{5} \\ - & 1 & 0 & 0 & 3 \\ 0 & - & 0 & 0 & 2 \\ 0 & \mathbf{3} & - & 0 & \mathbf{5} \\ 0 & \mathbf{3} & 0 & - & \mathbf{5} \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^5 = \begin{bmatrix} \mathbf{2} & 0 & - \\ \mathbf{3} & \mathbf{3} & - \\ 0 & 0 & - \\ 0 & 0 & - \\ \mathbf{5} & \mathbf{5} & 3 \end{bmatrix}; \qquad S = \begin{bmatrix} \mathbf{3} & \mathbf{0} & 8 & 5 & \mathbf{4} \\ - & 0 & 6 & 3 & 3 \\ 1 & - & 6 & 6 & 0 \\ 6 & \mathbf{3} & - & 5 & \mathbf{2} \\ 3 & \mathbf{3} & 5 & - & \mathbf{3} \\ 6 & 2 & 7 & 8 & - \end{bmatrix}$$

$custo = 3$

$Inc(5) = \{0, 1, 2, 5\}, \quad Exc(5) = \{3, 4\}$

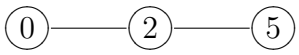
$Pos_5(0) = 1$

logo, $Pos_5 = \begin{bmatrix} 1 & 1 & 2 & - & - & 3 \end{bmatrix}$

1.5. $Dest = \{1, 2, 3, 4\}$

Ir para o passo 2.

2. O caminho encontrado para o nodo 5 foi:
Voltar ao passo 1.



Iteração nº4:

1.1. Seleccionamos o nodo $k = 4$, vamos agora determinar o caminho para o nodo 4.

$Inc(4) = \{4\}, \quad Exc(4) = \{0, 1, 2, 3, 5\}$

$Pos_4(4) = 3$

logo, $Pos_4 = \begin{bmatrix} - & - & - & - & 3 & - \end{bmatrix}$

As variáveis λ_{ij}^4 e γ_{jh}^4 foram inicializadas no passo 0, da seguinte forma:

$$\lambda^4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ - & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^4 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & 0 \\ 0 & 0 & - \end{bmatrix}$$

1.2. Tem-se que:

$\mathcal{A}(4) = \{(0, 4), (1, 4), (2, 4), (3, 4), (5, 4)\}$

$\mathcal{A}_3(4) = \emptyset$

logo, os arcos candidatos a incluir na solução são:

$\mathcal{A}(4) \backslash \mathcal{A}_3(4) = \{(0, 4), (1, 4), (2, 4), (3, 4), (5, 4)\}$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{04}, s_{14}, s_{24}, s_{34}, s_{54}\} = min\{5, 3, 6, 5, 8\} = 3 = s_{14}$

logo, vamos incluir o arco $(1, 4)$

1.4. As variáveis cujos valores vão ser alterados são:

$\lambda_{04}^4 = \lambda_{14}^4 = \lambda_{24}^4 = \lambda_{34}^4 = \lambda_{54}^4 = 3$

$\gamma_4^{14} = \gamma_4^{24} = \gamma_4^{34} = 3$

$s_{04} = 2, \quad s_{14} = 0, \quad s_{24} = 3, \quad s_{34} = 2, \quad s_{54} = 5$

assim,

$$\lambda^4 = \begin{bmatrix} 0 & 0 & 0 & \mathbf{3} & 0 \\ - & 0 & 0 & \mathbf{3} & 0 \\ 0 & - & 0 & \mathbf{3} & 0 \\ 0 & 0 & - & \mathbf{3} & 0 \\ 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & \mathbf{3} & - \end{bmatrix}; \qquad \gamma^4 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ \mathbf{3} & \mathbf{3} & \mathbf{3} \\ 0 & 0 & - \end{bmatrix}; \qquad S = \begin{bmatrix} 3 & 0 & 8 & \mathbf{2} & 4 \\ - & 0 & 6 & \mathbf{0} & 3 \\ 1 & - & 6 & \mathbf{3} & 0 \\ 6 & 3 & - & \mathbf{2} & 2 \\ 3 & 3 & 5 & - & 3 \\ 6 & 2 & 7 & \mathbf{5} & - \end{bmatrix}$$

$custo = 6$

$Inc(4) = \{1, 4\}, \quad Exc(4) = \{0, 2, 3, 5\}$

$Pos_4(1) = 2$

logo, $Pos_4 = \begin{bmatrix} - & 2 & - & - & 3 & - \end{bmatrix}$

1.5. Ir para o passo 1.2.

Iteração nº5:

1.2. Tem-se que:

$\mathcal{A}(4) = \{(0,1), (0,4), (2,1), (2,4), (3,1), (3,4), (5,1), (5,4)\}$

$\mathcal{A}_3(4) = \emptyset$

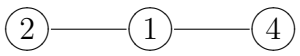
logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(4) \backslash \mathcal{A}_3(4) = \{(0,1), (0,4), (2,1), (2,4), (3,1), (3,4), (5,1), (5,4)$$

1.3. determinemos o valor de δ,

$\delta = min\{s_{01}, s_{04}, s_{21}, s_{24}, s_{31}, s_{34}, s_{51}, s_{54}\} = min\{3, 2, 1, 3, 6, 2, 6, 5\} = 1 = s_{21}$

logo, vamos incluir o arco $(2,1)$



1.4. As variáveis cujos valores vão ser alterados são:

$\lambda_{01}^4 = \lambda_{21}^4 = \lambda_{31}^4 = \lambda_{51}^4 = 1 \quad \lambda_{04}^4 = \lambda_{24}^4 = \lambda_{34}^4 = \lambda_{54}^4 = 4$

$\gamma_1^{14} = \gamma_1^{24} = 1$

$\gamma_4^{14} = \gamma_4^{24} = \gamma_4^{34} = 4$

$s_{01} = 2, \quad s_{21} = 0, \quad s_{31} = 5, \quad s_{51} = 5 \quad s_{04} = 1, \quad s_{24} = 2, \quad s_{34} = 1, \quad s_{54} = 4$

assim,

$$\lambda^4 = \begin{bmatrix} \mathbf{1} & 0 & 0 & \mathbf{4} & 0 \\ - & 0 & 0 & 3 & 0 \\ \mathbf{1} & - & 0 & \mathbf{4} & 0 \\ \mathbf{1} & 0 & - & \mathbf{4} & 0 \\ 0 & 0 & 0 & - & 0 \\ \mathbf{1} & 0 & 0 & \mathbf{4} & - \end{bmatrix}; \qquad \gamma^4 = \begin{bmatrix} \mathbf{1} & \mathbf{1} & - \\ 0 & 0 & - \\ 0 & 0 & - \\ \mathbf{4} & \mathbf{4} & \mathbf{4} \\ 0 & 0 & - \end{bmatrix}; \qquad S = \begin{bmatrix} \mathbf{2} & 0 & 8 & \mathbf{1} & 4 \\ - & 0 & 6 & 0 & 3 \\ \mathbf{0} & - & 6 & \mathbf{2} & 0 \\ \mathbf{5} & 3 & - & \mathbf{1} & 2 \\ 3 & 3 & 5 & - & 3 \\ \mathbf{5} & 2 & 7 & \mathbf{4} & - \end{bmatrix}$$

$custo = 7$

$Inc(4) = \{1, 2, 4\}, \quad Exc(4) = \{0, 3, 5\}$

$Pos_4(2) = 1$

Assim, $Pos_4 = \begin{bmatrix} - & 2 & 1 & - & 3 & - \end{bmatrix}$

1.5. Ir para o passo 1.2.

Iteração nº6:

1.2. Tem-se que:

$\mathcal{A}(4) = \{(0,1), (0,2), (0,4), (3,1), (3,2), (3,4), (5,1), (5,2), (5,4)\}$

$\mathcal{A}_3(4) = \{(3,2), (5,2)\}$

logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(4)\backslash\mathcal{A}_3(4) = \{(0,1),(0,2),(0,4),(3,1),(3,4),(5,1),(5,4)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{01}, s_{02}, s_{04}, s_{31}, s_{34}, s_{51}, s_{54}\} = min\{2,0,1,5,1,5,4\} = 0 = s_{02}$

logo, vamos incluir o arco $(0,2)$

1.4. Como $\delta = 0$ então não há alterações a realizar nos valores das matrizes λ^4, γ^4 e S

$custo = 7$

$Inc(4) = \{0,1,2,4\}, \quad Exc(4) = \{3,5\}$

$Pos_4(0) = 0$

logo, $Pos_4 = \begin{bmatrix} 0 & 2 & 1 & - & 3 & - \end{bmatrix}$

1.5. $Dest = \{1,2,3\}$

Ir para o passo 2.

2. O caminho encontrado para o nodo 4 foi:

Voltar ao passo 1.

Iteração nº7:

1.1. Seleccionamos o nodo $k = 3$, vamos agora determinar o caminho para o nodo 3.

$Inc(3) = \{3\}, \quad Exc(3) = \{0, 1, 2, 4, 5\}$

$Pos_3(3) = 3$

logo, $Pos_3 = \begin{bmatrix} - & - & - & 3 & - & - \end{bmatrix}$

As matrizes λ^3 e γ^3 foram inicializadas no passo 0, da seguinte forma:

$$\lambda^3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ - & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^3 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & 0 \\ 0 & 0 & - \\ 0 & 0 & - \end{bmatrix}$$

1.2. Tem-se que:

$\mathcal{A}(3) = \{(0,3), (1,3), (2,3), (4,3), (5,3)\}$

$\mathcal{A}_3(3) = \emptyset$

logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(3) \backslash \mathcal{A}_3(3) = \{(0,3), (1,3), (2,3), (4,3), (5,3)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{03}, s_{13}, s_{23}, s_{43}, s_{53}\} = min\{8, 6, 6, 5, 7\} = 5 = s_{43}$

logo, vamos incluir o arco $(4, 3)$

1.4. As variáveis cujos valores vão ser alterados são:

$\lambda^3_{03} = \lambda^3_{13} = \lambda^3_{23} = \lambda^3_{43} = \lambda^3_{53} = 5$

$\gamma^{13}_3 = \gamma^{23}_3 = \gamma^{33}_3 = 5$

$s_{03} = 3, \quad s_{13} = 1, \quad s_{23} = 1, \quad s_{43} = 0, \quad s_{53} = 2$

assim,

$$\lambda^3 = \begin{bmatrix} 0 & 0 & \mathbf{5} & 0 & 0 \\ - & 0 & \mathbf{5} & 0 & 0 \\ 0 & - & \mathbf{5} & 0 & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & \mathbf{5} & - & 0 \\ 0 & 0 & \mathbf{5} & 0 & - \end{bmatrix}; \qquad \gamma^3 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & - \\ \mathbf{5} & \mathbf{5} & \mathbf{5} \\ 0 & 0 & - \\ 0 & 0 & - \end{bmatrix}; \qquad S = \begin{bmatrix} 2 & 0 & \mathbf{3} & 1 & 4 \\ - & 0 & \mathbf{1} & 0 & 3 \\ 0 & - & \mathbf{1} & 2 & 0 \\ 5 & 3 & - & 1 & 2 \\ 3 & 3 & \mathbf{0} & - & 3 \\ 5 & 2 & \mathbf{2} & 4 & - \end{bmatrix}$$

$custo = 12$

$Inc(3) = \{3, 4\}, \quad Exc(3) = \{0, 1, 2, 5\}$

$Pos_3(4) = 2$

logo, $Pos_3 = \begin{bmatrix} - & - & - & 3 & 2 & - \end{bmatrix}$

1.5. Ir para o passo 1.2.

Iteração nº8:

1.2. Tem-se que:

$\mathcal{A}(3) = \{(0,3),(0,4),(1,3),(1,4),(2,3),(2,4),(5,3),(5,4)\}$

$\mathcal{A}_3(3) = \emptyset$

logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(3) \backslash \mathcal{A}_3(3) = \{(0,3),(0,4),(1,3),(1,4),(2,3),(2,4),(5,3),(5,4)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{03}, s_{04}, s_{13}, s_{14}, s_{23}, s_{24}, s_{53}, s_{54}\} = min\{3,1,1,0,1,2,2,4\} = 0 = s_{14}$

logo, vamos incluir o arco $(1,4)$

1.4. Como $\delta = 0$ então não vão existir alterações nos valores das matrizes λ^3, γ^3 e S

$custo = 12$
$Inc(3) = \{1, 3, 4\}, \quad Exc(3) = \{0, 2, 5\}$
$Pos_3(1) = 1$
logo, $Pos_3 = \begin{bmatrix} - & 1 & - & 3 & 2 & - \end{bmatrix}$

1.5. Ir para o passo 1.2.

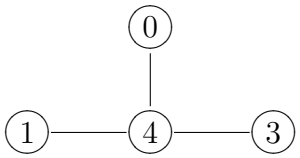
Iteração nº9:

1.2. Tem-se que:
$\mathcal{A}(3) = \{(0,1),(0,3),(0,4),(2,1),(2,3),(2,4),(5,1),(5,3),(5,4)\}$
$\mathcal{A}_3(3) = \{(2,1),(5,1)\}$

logo, os arcos candidatos a incluir na solução são:
$\mathcal{A}(3) \backslash \mathcal{A}_3(3) = \{(0,1),(0,3),(0,4),(2,3),(2,4),(5,3),(5,4)\}$

1.3. Determinemos o valor de δ,
$\delta = min\{s_{01}, s_{03}, s_{04}, s_{23}, s_{24}, s_{53}, s_{54}\} = min\{2,3,1,1,2,2,4\} = 1 = s_{04} = s_{23}$

Este valor ocorre para os arcos $(0,4)$ e $(2,3)$. Considere-se $(i_{aux}, j_{aux}) = (0,4)$, ou seja inclua-se o arco $(0,4)$



1.4. As variáveis cujos valores vão ser alterados são:
$\lambda^3_{01} = 1, \quad \lambda^3_{03} = \lambda^3_{23} = \lambda^3_{53} = 6, \quad \lambda^3_{04} = \lambda^3_{24} = \lambda^3_{54} = 1$
$\gamma^{13}_1 = 1, \quad \gamma^{13}_3 = \gamma^{23}_3 = 6, \quad \gamma^{13}_4 = \gamma^{23}_4 = 1$

$s_{01} = 1, \quad s_{03} = 2, \quad s_{23} = 0, \quad s_{53} = 1, \quad s_{04} = 0, \quad s_{24} = 1, \quad s_{54} = 3$

assim,

$$\lambda^3 = \begin{bmatrix} \mathbf{1} & 0 & \mathbf{6} & \mathbf{1} & 0 \\ - & 0 & 5 & 0 & 0 \\ 0 & - & \mathbf{6} & \mathbf{1} & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & 5 & - & 0 \\ 0 & 0 & \mathbf{6} & \mathbf{1} & - \end{bmatrix}; \qquad \gamma^3 = \begin{bmatrix} \mathbf{1} & 0 & - \\ 0 & 0 & - \\ \mathbf{6} & \mathbf{6} & 5 \\ \mathbf{1} & \mathbf{1} & - \\ 0 & 0 & - \end{bmatrix}; \qquad S = \begin{bmatrix} \mathbf{1} & 0 & \mathbf{2} & 0 & 4 \\ - & 0 & 1 & 0 & 3 \\ 0 & - & \mathbf{0} & \mathbf{1} & 0 \\ 5 & 3 & - & 1 & 2 \\ 3 & 3 & 0 & - & 3 \\ 5 & 2 & \mathbf{1} & \mathbf{3} & - \end{bmatrix}$$

$custo = 12$

$Inc(3) = \{0, 1, 3, 4\}, \quad Exc(3) = \{2, 5\}$

$Pos_3(0) = 1$

logo, $Pos_3 = \begin{bmatrix} 1 & 1 & - & 3 & 2 & - \end{bmatrix}$

1.5. $Dest = \{1, 2\}$

Ir para o passo 2.

2. O caminho encontrado para o nodo 3 foi:

Voltar ao passo 1.

Iteração nº10:

1.1. Seleccionamos o nodo $k = 2$. Vamos determinar o caminho para o nodo 2.

$Inc(2) = \{2\}, \quad Exc(2) = \{0, 1, 3, 4, 5\}$

$Pos_2(2) = 3$

logo, $Pos_2 = \begin{bmatrix} - & - & 3 & - & - & - \end{bmatrix}$

As matrizes λ^2 e γ^2 foram inicializadas no passo 0, da seguinte forma:

$$\lambda^2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ - & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^2 = \begin{bmatrix} 0 & 0 & - \\ 0 & 0 & 0 \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \end{bmatrix}$$

1.2. Tem-se que:

$\mathcal{A}(2) = \{(0,2),(1,2),(3,2),(4,2),(5,2)\}$

$\mathcal{A}_3(2) = \emptyset$

logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(2) \backslash \mathcal{A}_3(2) = \{(0,2),(1,2),(3,2),(4,2),(5,2)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{02}, s_{12}, s_{32}, s_{42}, s_{52}\} = min\{0,0,3,3,2\} = 0 = s_{02} = s_{12}$

Este valor ocorre para os arcos $(0,2)$ e $(1,2)$. Considere-se $(i_{aux}, j_{aux}) = (0,2)$ ou seja, incluia-se o arco $(0,2)$

1.4. Como $\delta = 0$ então não há alterações a realizar nos valores das matrizes λ^2, γ^2 e S

$custo = 12$

$Inc(2) = \{0,2\}, \quad Exc(2) = \{1,3,4,5\}$

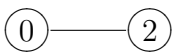
$Pos_2(0) = 2$

logo, $Pos_2 = \begin{bmatrix} 2 & - & 3 & - & - & - \end{bmatrix}$

1.5. $Dest = \{1\}$

Ir para o passo 2.

2. O caminho encontrado para o nodo 2 foi:



Voltar ao passo 1.

Iteração nº11:

1.1. $k = 1$. Resta-nos determinar o caminho para o nodo 1.

$Inc(1) = \{1\}, \quad Exc(1) = \{0, 2, 3, 4, 5\}$

$Pos_1(1) = 3$

logo, $Pos_1 = \begin{bmatrix} - & 3 & - & - & - & - \end{bmatrix}$

As matrizes λ^1 e γ^1 foram inicializadas no passo 0, da seguinte forma:

$$\lambda^1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ - & 0 & 0 & 0 & 0 \\ 0 & - & 0 & 0 & 0 \\ 0 & 0 & - & 0 & 0 \\ 0 & 0 & 0 & - & 0 \\ 0 & 0 & 0 & 0 & - \end{bmatrix}; \qquad \gamma^1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \\ 0 & 0 & - \end{bmatrix}$$

1.2. Tem-se que:

$\mathcal{A}(1) = \{(0,1), (2,1), (3,1), (4,1), (5,1)\}$

$\mathcal{A}_3(1) = \emptyset$

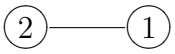
logo, os arcos candidatos a incluir na solução são:

$$\mathcal{A}(1) \backslash \mathcal{A}_3(1) = \{(0,1), (2,1), (3,1), (4,1), (5,1)\}$$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{01}, s_{21}, s_{31}, s_{41}, s_{51}\} = min\{1, 0, 5, 3, 5\} = 0 = s_{21}$

logo, vamos incluir o arco $(2, 1)$



1.4. Como $\delta = 0$ então não há alterações nos valores das matrizes λ^1, γ^1 e S

$custo = 12$

$Inc(1) = \{1, 2\}, \quad Exc(1) = \{0, 3, 4, 5\}$

$Pos_1(2) = 2$

logo, $Pos_1 = \begin{bmatrix} - & 3 & 2 & - & - & - \end{bmatrix}$

1.5. Ir para o passo 1.2.

Iteração nº12:

1.2. Tem-se que:

$\mathcal{A}(1) = \{(0,1), (0,2), (3,1), (3,2), (4,1), (4,2), (5,1), (5,2)\}$

$\mathcal{A}_3(1) = \emptyset$

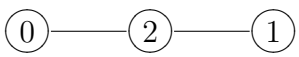
logo, os arcos candidatos a incluir na solução são:

$\mathcal{A}(1) \backslash \mathcal{A}_3(1) = \{0,1), (0,2), (3,1), (3,2), (4,1), (4,2), (5,1), (5,2)\}$

1.3. Determinemos o valor de δ,

$\delta = min\{s_{01}, s_{02}, s_{31}, s_{32}, s_{41}, s_{42}, s_{51}, s_{52}\} = min\{1, 0, 5, 3, 3, 3, 5, 2\} = 0 = s_{02}$

logo, vamos incluir o arco $(0, 2)$



1.4. Como $\delta = 0$ então não há alterações nos valores das matrizes λ^1, γ^1 e S

$custo = 12$

$Inc(1) = \{0, 1, 2\}, \quad Exc(1) = \{3, 4, 5\}$

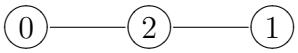
$Pos_1(0) = 1$

logo, $Pos_1 = \begin{bmatrix} 1 & 3 & 2 & - & - & - \end{bmatrix}$

1.5. $Dest = \emptyset$

Ir para o passo 2.

2. O caminho encontrado para o nodo 1 foi:



STOP

A solução dual ascendente é a que se apresenta na Figura 3.5 e tem custo 12.

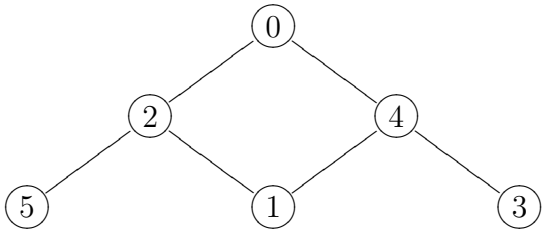


Figura 3.5: Solução dual ascendente

Terminamos este capítulo referindo que a solução óptima deste problema exemplo foi obtida usando uma formulação em programação linear inteira no CPLEX, tem custo 15 e é a que se apresenta na Figura 3.6. Notamos que a solução dual ascendente não contém a solução óptima.

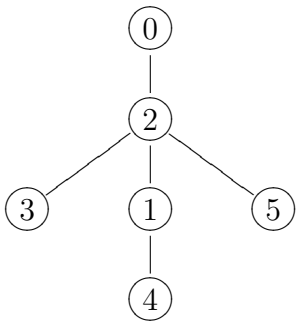


Figura 3.6: Solução óptima

Capítulo 4

Heurística Baseada na Solução Dual Ascendente

Neste capítulo apresentamos uma heurística que transforma a solução dual ascendente numa solução primal admissível, isto é, numa árvore de suporte com restrições de salto.

Como referimos no capítulo anterior a solução determinada pelo algoritmo Dual Ascendente, pode não ser uma solução primal admissível, no entanto contém uma. Assim, podemos determinar uma solução primal admissível eliminando arcos da solução dual admissível encontrada pelo algoritmo. De seguida descreveremos como é que o podemos fazer.

Para cada nodo $k \in \mathcal{D}$ o algoritmo Dual Ascendente obtém um caminho com restrição de salto da raíz para esse nodo k, e a solução dual ascendente obtém-se juntando todos os caminhos válidos encontrados. Observamos que, para se obter uma solução que contenha um caminho desde a raíz para todos os nodos destinos não precisamos de todos os caminhos válidos determinados pelo algoritmo. Consideremos um nodo k, se o caminho encontrado não coincidir com o caminho $[0, k]$, isto é, constituído apenas pelo arco $(0, k)$, então esse caminho contém caminhos válidos para outros nodos destino. Mais precisamente, esse caminho contém caminhos com restrição de salto para todos os nodos intermédios que pertencem ao caminho da raíz para esse nodo k. Portanto, uma vez incluído na solução o caminho para o nodo k, deixa de ser necessário incluir os caminhos determinados para os

nodos intermédios deste caminho. Considerando o exemplo de aplicação apresentado no capítulo anterior, o caminho encontrado para o nodo 3 foi $[0, 4, 3]$. Este caminho contém um caminho válido para o nodo 4, o caminho $[0, 4]$, portanto adicionando à solução o caminho encontrado para o nodo 3 já não é necessário incluir o caminho determinado para o nodo 4, isto é, não é necessário incluir o caminho $[0, 2, 1, 4]$.

Esta observação sugere que se determine uma solução utilizando o seguinte processo. Começamos com um nodo k e incluímos na solução o caminho determinado pelo algoritmo Dual Ascendente para esse nodo. Desta forma são também obtidos caminhos para os nodos intermédios desse caminho. Depois tomamos um dos restantes nodos, isto é, um nodo para qual ainda não foi determinado um caminho, e incluímos o caminho para esse nodo que é também um caminho para todos os nodos intermédios. Procedemos desta forma até termos obtido caminhos para todos os nodos.

Consideremos o exemplo apresentado no capítulo anterior, os caminhos encontrados foram: $[0, 2, 5]$, $[0, 2, 1, 4]$, $[0, 4, 3]$, $[0, 2]$ e $[0, 2, 1]$. Para determinar uma solução da forma atrás descrita consideramos os caminhos pela ordem inversa à qual foram determinados, ou seja, começamos com o último caminho encontrado. Introduz-se esse caminho e eliminamos todos os caminhos para nodos intermédios desse caminho. Prosseguimos com o próximo caminho da lista não eliminado. Assim, introduzimos o caminho $[0, 2, 1]$, este caminho é também um caminho para o nodo 2. Antes do nodo 1 foi pesquisado o nodo 2, para o qual já temos um caminho e portanto não precisamos de incluir o caminho determinado pelo algoritmo (que neste caso, até coincide com o caminho já introduzido). Antes do nodo 2 foi pesquisado o nodo 3, para o qual ainda não temos um caminho, portanto introduz-se o caminho determinado pelo algoritmo, ou seja, adicionamos o caminho $[0, 4, 3]$. Este caminho é também um caminho para o nodo 4 (pelo que já não é necessário introduzir o caminho $[0, 2, 1, 4]$). Antes do nodo 3 foi pesquisado o nodo 4, para o qual já temos um caminho. Antes do nodo 4 foi pesquisado o nodo 5, para o qual ainda não temos um caminho, portanto adicionamos o caminho $[0, 2, 5]$. Já determinámos um caminho para todos os nodos destinos, pelo que a solução encontrada foi a que se apresenta na Figura 4.1.

Esta solução é uma árvore de suporte com restrição de salto e como tal é uma solução

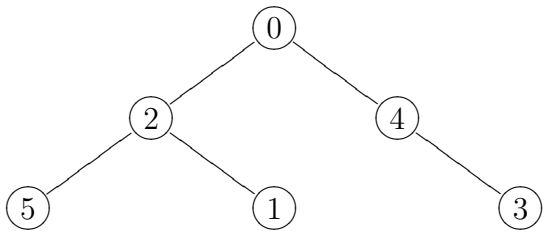


Figura 4.1: Solução heurística primal admissível

primal admissível. No entanto notamos que se tivessemos começado pelo primeiro caminho encontrado, obteríamos a solução que se encontra na Figura 4.2. Esta solução coincide com a solução dual ascendente que já vimos não ser uma solução primal admissível.

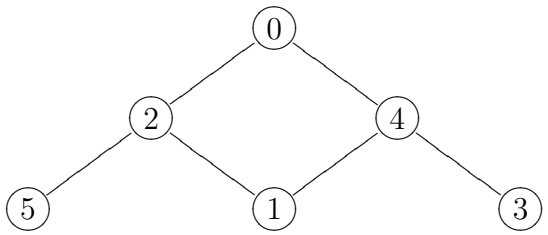


Figura 4.2: Solução heurística primal não admissível

Este exemplo mostra que este procedimento por vezes é suficiente para obter uma solução primal admissível. Assim, o que acabámos de descrever constitui o primeiro passo da heurística que construímos. Após este passo verificamos se a solução é primal admissível, se o for paramos, se não procedemos como se descreve de seguida.

Se a solução não é primal admissível isso significa que existem ciclos. Assim, o próximo passo da heurística começa por identificar esses ciclos. Após identificarmos o ciclo a eliminar temos de determinar quais as arestas que podemos eliminar, por forma a garantir que a árvore que se obtém verifica as restrições de salto. No exemplo anterior existe o ciclo $[0, 2, 1, 4, 0]$, a aresta $\{0, 2\}$ não pode ser eliminada pois se o fizessemos o caminho da raíz para o nodo 5, $[0, 4, 1, 2, 5]$, teria 4 arestas e violava as restrições de salto. Pelo mesmo motivo não podemos eliminar a aresta $\{0, 4\}$ pois se o fizessemos o caminho da raíz para

o nodo 3, $[0, 2, 1, 4, 3]$, não verificava as restrições de salto. Assim, por forma a eliminar o ciclo e obtermos uma solução primal admissível temos de eliminar uma das arestas $\{2, 1\}$ ou $\{4, 1\}$.

Notamos que se existe um ciclo na solução então na correspondente rede orientada existe um nodo com mais do que um caminho orientado da raíz para esse nodo. Assim, para identificarmos os ciclos consideramos a forma orientada da solução obtida no passo 1 e verificamos se existe mais do que um caminho orientado para algum nodo k. Se assim acontecer, uma forma de eliminar esses ciclos consiste em eliminar todos os arcos incidentes nesse nodo menos um. Por forma a garantir que as restrições de salto são verificadas eliminamos os arcos que provêm dos caminhos, para esse nodo, com mais arcos. Procedendo desta forma eliminamos todos os ciclos existentes e garantimos que a solução é uma árvore e que só contém caminhos válidos, ou seja, garantimos a obtenção de uma solução primal admissível.

Consideremos novamente o exemplo anterior. Apresentamos na Figura 4.3 a solução orientada da solução dual ascendente que apresentámos na Figura 4.2.

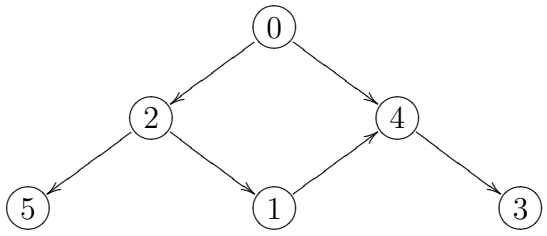


Figura 4.3: Solução dual ascendente orientada

Existem dois caminhos orientados para o nodo 4, pois é o único nodo com grau de entrada superior a um, o que significa que existe um ciclo (não orientado) que inclui este nodo. Os caminhos orientados para o nodo 4 são o caminho $[0, 4]$ com apenas um arco e o caminho $[0, 2, 1, 4]$ com três arcos. Por forma a eliminar o ciclo ou eliminamos o arco $(0, 4)$ ou o arco $(1, 4)$. Uma vez que o arco $(1, 4)$ faz parte do caminho orientado para o

nodo 4 com mais arcos, é este o arco que é eliminado da solução. Removendo a orientação dos arcos obtemos uma solução primal admissível que, neste caso, coincide com a solução apresentada na Figura 4.1.

Em seguida apresentamos a notação que utilizámos para descrever esta heuristica.

Relembramos que D representa o conjunto dos nodos destinos, n o número de elementos deste conjunto e c_{ij} o custo do arco (i, j).

Representaremos por $caminho(k)$ o conjunto formado por todos os arcos pertencentes ao caminho da raíz para o nodo k na solução dual ascendente.

Utilizaremos a matriz *solheur* para indicar a solução construída com esta heurística e a variável *custo* para representar o custo dessa solução. Assim,

$$solheur(i,j) = \begin{cases} 1 & \text{, se o arco } (i,j) \text{ pertence à solução} \\ 0 & \text{, caso contrário} \end{cases} , \forall (i,j) \in \mathcal{A}.$$

Utilizaremos o conjunto *Dest* para representar o conjunto dos nodos para os quais já foi identificado um caminho.

Utilizaremos ainda, as variáveis $p(k)$ que indicam o número de arcos incidentes em k na solução construída pela heurística, representaremos por i_1^k, i_2^k, ..., $i_{p(k)}^k$ os nodos incidentes em k e por $comp(i_l^k)$ o comprimento do maior caminho orientadado para k que passa no nodo i_l^k.

A heurística pode então ser descrita da seguinte forma.

Passo 0. Inicialização

Fazer: $Dest = \emptyset$

$solheur(i,j) = 0, \quad \forall i = 0,..,n; j = 1,..,n$

$custo = 0$

Passo 1. Construção da solução

Repetir as seguintes instruções até que $Dest = \mathcal{D}$

Seleccionar um nodo destino $k \in \mathcal{D} \backslash Dest$

$i = 0$

Repetir as seguintes instruções enquanto $i \neq k$

selecionar o arco $(i, j) \in caminho(k)$ e fazer: $solheur(i, j) = 1$

$custo = custo + c_{ij}$

$Dest = Dest \cup \{j\}$

$i = j$

Passo 2. Condição de paragem

Para cada $k \in \mathcal{D}$ determinar o número de arcos incidentes nesse nodo,

$$p(k) = \sum_{i=0}^{n} solheur(i, k).$$

Se $p(k) = 1$ para todo o $k \in D$ então Stop, caso contrário ir para o passo 3.

Passo 3. Eliminação dos ciclos

Para cada $k \in \mathcal{D}$ tal que $p(k) > 1$ determinar o comprimento de todos os caminhos orientados para k, $comp(i_l^k)$ com $l = 1, ..., p(k)$.

Fazer: $aux = comp(i_{1k})$

$i_{aux} = i_1^k$

$l = 2$

Repetir as seguintes instruções enquanto $l \leq p(k)$

Se $comp(i_l^k) > aux$ ou $(comp(i_l^k) = aux$ e $c_{i_l^k,k} \geq c_{i_{aux},k})$

então $solheur(i_l^k, k) = 0$

$custo = custo - c_{i_l^k,k}$

$l = l + 1$

Se $comp(i_l^k) < aux$ ou $(comp(i_l^k) = aux$ e $c_{i_l^k,k} < c_{i_{aux},k})$

então $solheur(i_{aux}, k) = 0$

$$custo = custo - c_{i_{aux},k}$$

$$i_{aux} = i_l^k$$

$$aux = comp(i_l^k)$$

$$l = l + 1$$

A solução que se obtém no final é uma solução primal admissível e o seu valor representa um limite superior para o valor óptimo do problema.

No exemplo que temos vindo a utilizar, recorde que a matriz de custos é:

$$C = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \left[\begin{array}{cccccc} - & 5 & 3 & 8 & 5 & 9 \\ 5 & - & 1 & 6 & 3 & 6 \\ 3 & 1 & - & 6 & 6 & 2 \\ 8 & 6 & 6 & - & 5 & 7 \\ 5 & 3 & 6 & 5 & - & 8 \\ 9 & 6 & 2 & 7 & 8 & - \end{array}\right] \end{array}.$$

O valor da solução apresentada na Figura 4.1 é 16, que correponde a um limite superior do valor óptimo da solução que é 15 representada na Figura 3.6.

Capítulo 5

Testes Computacionais

Neste capítulo apresentamos alguns resultados computacionais obtidos pelo algoritmo Dual Ascendente apresentado no Capítulo 3 e pela heurística apresentada no Capítulo 4 usando instâncias do problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto. As instâncias que usamos correspondem a grafos completos com custos euclideanos e aleatórios e com um número de nodos destino igual a 20, 40 e 60 e com o valor do parâmetro H igual a 3, 4 e 5.

Para se obterem as instâncias com custos euclideanos foram geradas aleatoriamente as coordenadas de $n+1$ pontos (que correspondem aos nodos do grafo) numa grelha de dimensão 100×100. Posteriormente o custo, c_{ij}, de cada aresta $\{i, j\}$ foi obtido tomando a parte inteira da distância euclideana entre os pontos i e j. Para as distâncias euclideanas foram consideradas duas localizações para o nodo raíz, uma em que o nodo raíz se encontra no centro da grelha e outra em que a raíz se encontra num extremo da grelha, estas instâncias foram designadas, repectivamente, por TC e TE. O custo c_{ij}, de cada aresta $\{i, j\}$ das instâncias aleatórias foi obtido utilizando uma distribuição $U\,[0, 100]$. Para estas instâncias considerou-se que o nodo raíz era o nodo 0 e denotamos estas instâncias por TR. Todas estas instâncias têm sido usadas com frequência desde Gouveia[10] para a obtenção de resultados computacionais para este e outros problemas e os detalhes da forma como foram geradas pode ser consultada em Dahl et al.[5].

O Algoritmo Dual Ascendente e a heurística baseada na solução dual ascendente foram implementados computacionalmente utilizando a linguagem de programação FORTRAN. Os resultados computacionais foram obtidos utilizando um HP530 Centrino Duo, 1.66GHz com 1015MB de RAM e apresentam-se nas Tabelas 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8 e 5.9. Na primeira coluna destas tabelas, designada por PROB, identifica-se o problema (TC, TE ou TR) e indica-se o número de nodos destino do correspondente grafo (20,40 ou 60). Na segunda coluna, designada por H, indica-se o valor do parâmetro H considerado (3, 4 ou 5). Na terceira coluna, designada por OPT, encontra-se o valor da correspondente solução óptima ou um seu limite superior ou inferior (estes casos serão assinalados colocando, respectivamente, os símbolo (ls) ou (li) junto do valor). Os valores apresentados nesta coluna foram obtidos usando as formulações apresentadas no Capítulo 2 e mais detalhes sobre a forma como foram obtidos pode ser consultada em Dahl et al.[5]. As três colunas seguintes referem-se aos resultados obtidos pelo algoritmo Dual Ascendente. Na primeira dessas colunas, denotada por *valor*, colocou-se o valor da solução dual ascendente, na segunda, denotada por *tempo*, o tempo de execução do algoritmo em segundos e na terceira dessas colunas o valor do *gap* correspondente. O valor do gap determinou-se utilizando a seguinte fórmula $gap = \frac{OPT-valor}{OPT} \times 100$ e permite-nos ter uma noção da qualidade do limite inferior obtido com a aplicação do algoritmo Dual Ascendente. As últimas três colunas dizem respeito aos resultados obtidos pela aplicação da heurística baseada na solução dual ascendente. Na primeria dessas colunas, denotada por *valor*, colocou-se o valor da solução obtida, na segunda, denotada por *tempo*, o tempo de execução em segundos e na terceira dessas colunas o valor do *gap* correspondente que se determinou utilizando a seguinte fórmula $gap = \frac{valor-OPT}{OPT} \times 100$ e permite-nos ter uma noção da qualidade do limite superior obtido com a aplicação da heurística. Nos casos em que não se dispõe do valor óptimo, para calcular o valor do gap correspondente, utilizou-se para o Algoritmo Dual Ascendente um seu limite superior e para a heurística um limite inferior.

Por observação destas tabelas é imediato verificar que quer o algoritmo Dual Ascendente quer a heurística baseada na solução dual ascendente permitem de uma forma muito rápida obter, respectivamente, um limite inferior e um limite superior para o valor óptimo do problema. Notamos que a instância de teste TR3,60 com $H = 3$ (ver Tabela 5.9) é a

que apresenta maior tempo de execução sendo este valor de 0.07 segundos e que para as instâncias de teste com 20 nodos destino, à excepção de TE4,20 com $H = 4$ e TR3,20 com $H = 3$ e $H = 5$, os tempos computacionais são praticamente nulos (ver Tabelas 5.1, 5.2 e 5.3). No entanto, os valores dos *gaps* (por serem muito elevados) mostram que estes limites são de fraca qualidade. Notamos que de todos os testes efectuados, existem 10 instâncias com valor do gap superior a 113 sendo 218.36 o maior destes valores, este valor é referente à solução da heurística da instância TE1,60 com $H = 3$ (ver Tabela 5.8).

Também se verifica que em nenhuma das instâncias de teste foi possível determinar o seu valor óptimo, notamos que os valores obtidos quer com a aplicação do algoritmo Dual Ascendente quer com a heurística nunca coincidem com o valor óptimo apresentado.

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	340	248	0.00	27.06	422	0.00	24.12
TC1,20	4	262	262	0.00	17.61	416	0.04	30.82
	5	312	262	0.00	16.03	416	0.00	33.33
	3	365	280	0.00	23.29	479	0.00	31.23
TC2,20	4	338	293	0.00	13.31	432	0.00	27.81
	5	332	293	0.00	11.75	432	0.00	30.12
	3	343	252	0.00	26.53	422	0.00	23.03
TC3,20	4	306	260	0.00	15.03	425	0.00	38.89
	5	296	260	0.00	12.16	401	0.00	35.47
	3	390	279	0.00	28.46	588	0.02	50.77
TC4,20	4	376	291	0.00	22.61	565	0.00	50.27
	5	364	291	0.00	20.05	565	0.00	55.22
	3	347	255	0.00	26.51	463	0.00	33.43
TC5,20	4	326	281	0.00	13.80	461	0.00	41.41
	5	310	281	0.00	9.35	449	0.00	44.84

Tabela 5.1: Resultados computacionais para instâncias com custos euclideanos e raíz no centro da grelha com 20 nodos destino

Para as instâncias com custos euclideanos e raíz no centro da grelha com 20 nodos destino (ver Tabela 5.1) o melhor limite inferior foi obtido para a instância TC5 com $H = 5$ e a diferença entre o valor óptimo e o valor da solução dual ascendente é de 29 unidades a que corresponde um gap de 9.35. O pior limite inferior foi obtido para a instância TC4 com $H = 3$ sendo a diferença neste caso de 111 unidades ao qual corresponde um gap de

28.46. O pior limite superior foi obtido para a instância TC4 com $H = 5$, sendo a diferença entre o valor da solução obtida pela heurística e o valor óptimo de 201 unidades, a que corresponde um gap de 55.22. O melhor limite superior foi obtido para a instância TC3 com $H = 3$, sendo neste caso o gap de 23.03 que corresponde a uma diferença de 79 unidades. Notamos que para estes problemas as menores e as maiores diferenças observadas correspondem, respectivamente, aos melhores e aos piores limites encontrados.

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	449	215	0.00	52.12	694	0.01	54.57
TE1,20	4	385	263	0.00	31.69	555	0.02	44.16
	5	366	266	0.00	27.32	570	0.00	55.74
	3	435	164	0.00	62.30	1001	0.00	130.11
TE2,20	4	404	144	0.00	64.36	736	0.00	82.18
	5	383	190	0.00	50.39	687	0.00	79.37
	3	435	262	0.00	39.77	677	0.00	55.63
TE3,20	4	396	294	0.00	25.76	642	0.00	62.12
	5	372	295	0.00	20.70	606	0.01	62.90
	3	448	205	0.00	54.24	738	0.00	64.73
TE4,20	4	402	257	0.02	36.07	737	0.02	83.33
	5	382	257	0.00	32.72	737	0.00	92.93
	3	428	188	0.00	56.07	709	0.00	65.65
TE5,20	4	376	215	0.00	42.82	695	0.00	84.84
	5	354	233	0.00	34.18	674	0.00	90.40

Tabela 5.2: Resultados computacionais para instâncias com custos euclideanos e raíz no extremo da grelha com 20 nodos destino

Para as instâncias com custos euclideanos e raíz no extremo da grelha com 20 nodos destino (ver Tabela 5.2) e relativamente aos resultados obtidos com a aplicação do algoritmo Dual Ascendente, notamos que o melhor limite inferior, que corresponde à menor diferença encontrada entre o valor óptimo e o valor da solução dual ascendente é de 77 unidades para a instância TE3 com $H = 5$, sendo o gap neste caso de 20.70. A maior diferença foi verificada para a instância TE2 com $H = 3$, sendo a diferença de 271 unidades a que corresponde um gap de 62.30. No entanto, o maior gap, de 64.36, foi verificado para esta instância mas com $H = 4$, sendo neste caso a diferença de 260 unidades. Este último caso é o pior resultado obtido com a aplicação do algoritmo Dual Ascendente para

as instâncias com 20 nodos destino. No que diz respeito à heurística, a menor diferença encontrada entre o valor da solução obtida pela heurística e o valor óptimo do problema é de 170 unidades, a que corresponde um gap de 44.16 para a instância TE1 com $H = 4$ e a maior diferença foi verificada para a instância TE2 com $H = 3$, sendo neste caso a diferença de 565 com um gap de 130.11. Este último caso corresponde ao pior limite superior obtido com a aplicação da heurística para as instâncias com 20 nodos destino.

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	168	117	0.00	30.36	195	0.00	16.07
TR1,20	4	146	126	0.00	13.70	158	0.00	8.22
	5	137	126	0.00	8.03	158	0.00	15.33
	3	201	83	0.00	58.71	247	0.00	22.89
TR2,20	4	161	87	0.00	45.96	196	0.00	21.74
	5	140	96	0.00	31.43	169	0.00	20.71
	3	157	101	0.03	35.67	173	0.03	10.19
TR3,20	4	130	112	0.00	13.85	149	0.00	14.62
	5	121	116	0.01	4.13	126	0.01	4.13
	3	183	68	0.00	62.84	222	0.00	21.31
TR4,20	4	158	91	0.00	42.41	194	0.00	22.78
	5	142	92	0.00	35.21	175	0.00	23.24
	3	148	67	0.00	54.73	177	0.00	19.59
TR5,20	4	121	66	0.00	45.45	164	0.00	35.54
	5	110	74	0.00	32.73	129	0.00	17.27

Tabela 5.3: Resultados computacionais para instâncias com custos aleatórios com 20 nodos destino

Para as instâncias com custos aleatórios com 20 nodos destino (ver Tabela 5.3) os melhores limites inferior e superior foram verificados para a instância TR3 com $H = 5$ sendo as diferenças entre o valor óptimo e o valor da solução encontrada, em ambos os casos, de 5 unidades a que corresponde um gap de 4.13. Notamos que estes foram os melhores limites obtidos de entre todos os testes efectuados. A instância TR2 com $H = 3$ foi a que apresentou a maior diferença entre o valor óptimo e o valor da solução dual ascendente, sendo esta diferença de 118 unidades a que corresponde um gap de 58.71, no entanto o pior limite inferior encontrado corresponde à instância TR4 com $H = 3$, a diferença verificada foi de 115 unidades, a que corresponde um gap de 62.84. No que diz respeito à heurística,

o pior limite superior foi verificado para a instância TR5 com $H = 4$, neste caso o valor do gap é de 35.53 o qual corresponde a uma diferença entre o valor da solução obtida pela heurística e o valor óptimo de 33 unidades. A maior diferença observada entre estes valores é de 46 unidades para a instância TR2 com $H = 3$ e corresponde a um gap de 22.89.

Para as instâncias com 20 nodos destino, e analisando os resultados obtidos com a aplicação do algoritmo Dual Ascendente observamos que, no geral, os problemas com custos euclideanos e raíz no centro da grelha (TC) são os de mais fácil resolução, pois são os que, no geral, apresentam menores valores para os gaps e os problemas com custos euclideanos e raíz no extremo da grelha (TE) são os de mais difícil resolução, pois são os que apresentam maiores valores para os gaps. Os resultados obtidos com a aplicação da heurística revelam que, no geral, os problemas com custos euclideanos e raíz no extremo da grelha (TE) são também os de mais difícil resolução, sendo os problemas com custos aleatórios (TR) os de mais fácil resolução.

Quando comparamos os resultados do algoritmo Dual Ascendente com os da heurística, notamos que os valores dos gaps dos problemas com custos euclideanos (TC e TE) são normalmente superiores para a heurística, significando isto que os limites inferiores obtidos para estes problemas são melhores do que os limites superiores. Para os problemas com custos aleatórios (TR) verifica-se a relação contrária, isto é, os limite inferiores são, em geral, piores que os limites superiores.

Para as instâncias com custos euclideanos e raíz no centro da grelha com 40 nodos destino (ver Tabela 5.4) a menor diferença verificada entre o valor óptimo e o valor da solução dual ascendente é de 141 unidades para a instância TC2 com $H = 5$ e a maior diferença é de 367 unidades para a instância TC1 com $H = 3$ às quais corresponde um gap de 28.43 e de 60.26, respectivamente. Estes valores correspondem, respectivamente, ao melhor e ao pior limite inferior registado para estas instâncias. Quanto aos limites superiores, notamos que o melhor e o pior limite correspondem também, respectivamente, à menor e à maior diferença observada entre o valor da solução obtida pela heurística e o valor óptimo. A menor diferença é de 207 unidades para a instância TC2 com $H = 3$, a que corresponde um gap de 36.57. A maior diferença foi registada para a instância TC1 com $H = 4$, sendo

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	609	242	0.03	60.26	949	0.03	55.83
TC1,40	4	548	293	0.00	46.53	925	0.02	68.80
	5	522	312	0.02	40.23	844	0.02	61.69
	3	566	299	0.01	47.17	773	0.02	36.57
TC2,40	4	519	354	0.00	31.79	786	0.00	51.45
	5	496	355	0.02	28.43	728	0.02	46.77
	3	580	303	0.02	47.76	832	0.02	43.45
TC3,40	4	544	347	0.02	36.21	774	0.02	42.28
	5	516	349	0.01	32.36	762	0.01	47.67
	3	613	319	0.01	47.96	901	0.02	46.98
TC4,40	4	557	358	0.01	35.73	836	0.02	50.09
	5	524	364	0.02	30.53	767	0.04	46.37
	3	599	350	0.04	41.57	841	0.04	40.40
TC5,40	4	552	359	0.02	34.96	774	0.02	40.22
	5	522	359	0.00	31.23	801	0.00	53.45

Tabela 5.4: Resultados computacionais para instâncias com custos euclideanos e raíz no centro da grelha com 40 nodos destino

neste caso a diferença de 377 unidades a que corresponde um gap de 68.80.

Para as instâncias com custos euclideanos e raíz no extremo da grelha com 40 nodos destino (ver Tabela 5.5) o melhor limite inferior foi obtido para a instância de teste TE4 com $H = 5$ e a diferença entre o valor óptimo e o valor da solução dual ascendente é de 314 unidades a que corresponde um gap de 53.68. Notamos, no entanto, que a menor diferença observada é de 290 unidades para a instância TE3 com $H = 5$, a que corresponde um gap de 53.70. O pior limite inferior foi obtido para a instância TE2 com $H = 3$ sendo a diferença neste caso de 546 unidades ao qual corresponde um gap de 76.90, sendo esta a maior diferença observada e o pior resultado obtido com a aplicação o Algoritmo Dual Ascendente para as instâncias com 40 nodos destino. A menor diferença verificada entre o valor da solução obtida pela heurística e o valor óptimo é de 407 unidades, a que corresponde um gap de 65.12, valores estes obtidos para a instância TE4 com $H = 4$ e que corresponde ao melhor limite superior observado para estas instâncias. O pior limite superior foi obtido para a instância TE1 com $H = 3$, sendo neste caso o gap de 144.07 que corresponde à diferença máxima de 1020 unidades, sendo este caso o pior resultado obtido

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	708	243	0.00	65.68	1728	0.01	144.07
TE1,40	4	627	162	0.00	74.16	1526	0.01	143.38
	5	590	180	0.00	69.49	1386	0.01	134.92
	3	710	164	0.03	76.90	1393	0.03	96.20
TE2,40	4	625	173	0.02	72.32	1208	0.02	93.28
	5	581	233	0.01	59.90	1144	0.01	96.90
	3	660	240	0.01	63.64	1193	0.01	80.76
TE3,40	4	581	200	0.00	65.58	1013	0.03	74.35
	5	540	250	0.01	53.70	951	0.02	76.11
	3	722	235	0.00	67.45	1257	0.00	74.10
TE4,40	4	625	261	0.02	58.24	1032	0.03	65.12
	5	585	271	0.01	53.68	998	0.02	70.60
	3	675	191	0.00	71.70	1345	0.00	99.26
TE5,40	4	614	205	0.03	66.61	1226	0.03	99.67
	5	571	236	0.01	58.67	1221	0.02	113.84

Tabela 5.5: Resultados computacionais para instâncias com custos euclideanos e raíz no extremo da grelha com 40 nodos destino

com a aplicação da heurística para as instâncias com 40 nodos destino.

Os melhores resultados obtidos quer com a aplicação do algoritmo Dual Ascendente quer com a aplicação da heurística para as instâncias com 40 nodos destino foram verificados para instâncias com custos aleatórios (ver Tabela 5.6). O melhor limite inferior foi obtido para a instância TR1 com $H = 5$, neste caso verificou-se uma diferença de 36 unidades entre o valor da solução dual ascendente e o valor optimo, correspondendo a um gap de 25.90. O melhor limite superior foi obtido para a instância TR2 com $H = 5$, a diferença entre o valor da solução obtida pela heurística e o valor óptimo é de 20 unidades, a que corresponde um gap de 12.90. No entanto a menor destas diferenças é de 19 unidades para a instância TR4 com $H = 4$ à qual corresponde um gap de 14.39. Analisando ainda os resultados das instâncias com custos aleatórios com 40 nodos destino, notamos que o pior limite inferior, ao qual corresponde a diferença máxima, verificou-se para a instância TR2 com $H = 3$. A diferença entre os valores foi de 142 unidades a que corresponde um gap de 64.84. Para a heurística o gap mais elevado foi de 37.88 que corresponde à diferença máxima de 75 unidades verificada para a instância TR3 com $H = 3$.

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	176	78	0.02	55.68	213	0.02	21.02
TR1,40	4	149	92	0.01	38.26	187	0.01	25.50
	5	139	103	0.02	25.90	162	0.05	16.55
	3	219	77	0.01	64.84	286	0.01	30.59
TR2,40	4	176	78	0.02	55.68	217	0.02	23.30
	5	155	87	0.01	43.87	175	0.01	12.90
	3	198	72	0.01	63.64	273	0.01	37.88
TR3,40	4	145	69	0.00	52.41	171	0.02	17.93
	5	123	84	0.03	31.71	146	0.03	18.70
	3	167	67	0.00	59.88	223	0.02	33.53
TR4,40	4	132	79	0.04	40.15	151	0.04	14.36
	5	122	85	0.02	30.33	157	0.02	28.69
	3	205	100	0.02	51.22	246	0.02	20.00
TR5,40	4	159	100	0.00	37.11	190	0.00	19.50
	5	149	107	0.01	28.19	183	0.02	22.82

Tabela 5.6: Resultados computacionais para instâncias com custos aleatórios com 40 nodos destino

Analisando os resultados das instâncias com 40 nodos destino, concluímos, à semelhança do que se observou para as instâncias com 20 nodos destino, que, no que diz respeito aos resultados obtidos com a aplicação do Algoritmo Dual Ascendente, os problemas com custos euclideanos e raíz no centro da grelha (TC) são os de mais fácil resolução, enquanto que os problemas com custos euclideanos e raíz no extremo da grelha (TE) são os de mais difícil resolução. No que se refere aos resultados obtidos com a aplicação da heurística, notamos que os problemas com custos euclideanos e raíz no extremo da grelha (TE) são os que apresentam os maiores valores para os gaps e os problemas com custos aleatórios (TR) os que apresentam os menores valores.

Também neste caso se verifica que, no geral, os limites inferiores obtidos para os problemas com custos euclideanos (TC e TE) são melhores do que os limite superiores e que para os problemas com custos aleatórios se verifica a relação contrária.

Para as instâncias com custos euclideanos e raíz no centro da grelha com 60 nodos destino (ver Tabela 5.7) a menor diferença verificada entre o valor óptimo e o valor da solução

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	866	298	0.04	65.59	1378	0.04	59.12
TC1,60	4	781	318	0.03	59.28	1223	0.03	56.59
	5	734	351	0.04	52.18	1140	0.04	55.31
	3	845	364	0.05	56.92	1377	0.05	62.96
TC2,60	4	767	397	0.05	48.24	1156	0.05	50.72
	5	727	419	0.02	42.37	1127	0.03	55.02
	3	887	346	0.01	60.99	1463	0.02	64.94
TC3,60	4	804	387	0.01	51.87	1283	0.01	59.58
	5	764	396	0.01	48.17	1202	0.03	57.33
	3	855	364	0.04	57.43	1316	0.04	53.92
TC4,60	4	778	405	0.03	47.94	1236	0.05	58.87
	5	732	409	0.03	44.13	1186	0.03	62.02
	3	1000	411	0.03	58.90	1561	0.03	56.10
TC5,60	4	905	448	0.03	50.50	1394	0.05	54.03
	5	860	465	0.04	45.93	1374	0.04	59.77

Tabela 5.7: Resultados computacionais para instâncias com custos euclideanos e raíz no centro da grelha com 60 nodos destino

dual ascendente, que corresponde ao melhor limite inferior observado, é de 308 unidades, para a instância TC2 com $H = 5$, a que corresponde um gap de 42.37. A maior destas diferenças é de 589 unidades a que corresponde um gap de 58.90, verificada para a instância TC5 com $H = 3$. No entanto, foi para a instância TC1 com $H = 3$ que se verificou o pior limite inferior, sendo neste caso a diferença de 568 unidades a que corresponde um gap de 65.59. Relativamente à heurística o menor valor verificado entre o valor da solução obtida pela heurística e o valor óptimo é de 389 unidades, a que corresponde um gap de 50.72, para a instância TC2 com $H = 4$. A maior diferença entre estes valores registou-se para a instância TC3 com $H = 3$ e foi de 576 unidades, a que corresponde um gap de 64.94. Estes dois últimos resultados foram, respectivamente, o melhor e o pior limite superior observado para as instâncias com custos euclideanos e raíz no centro da grelha com 60 nodos destino.

Para as instâncias com custos euclideanos e raíz no extremo da grelha com 60 nodos destino (ver Tabela 5.8) o melhor limite inferior registado foi para a instância TE2 com $H = 5$ e apresenta uma diferença entre o limite superior do valor óptimo apresentado e o valor da solução dual ascendente de 783 unidades, ao qual corresponde um gap de 64.23.

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	1525	344	0.03	77.44	4855	0.03	218.36
TE1,60	4	1345(ls)	273	0.03	79.70			
		1315(li)				3672	0.03	179.24
	5	1238(ls)	319	0.01	74.23			
		1213(li)				3292	0.03	171.39
	3	1463	422	0.03	71.16	3451	0.03	135.89
TE2,60	4	1326(ls)	368	0.06	72.25	2650	0.06	620.11*
	5	1219(ls)	436	0.05	64.23	2470	0.05	466.51*
	3	1459(ls)	371	0.05	74.57			
		1430(li)				3743	0.05	161.57
TE3,60	4	1271(ls)	354	0.06	72.15	2752	0.06	677.40*
	5	1178(ls)	416	0,03	64.69	2695	0.05	547.84*

Tabela 5.8: Resultados computacionais para instâncias com custos euclideanos e raíz no extremo da grelha com 60 nodos destino

O pior limite inferior, que coincide também com o pior resultado do algoritmo Dual Ascendente de entre todas as instâncias testadas, corresponde à instância TE1 com $H = 4$, sendo a diferença neste caso de 1072 unidades, ao qual corresponde um gap de 79.70. Notamos, no entanto, que a menor e a maior diferença observada foi de 762 e 1181, para as instâncias TE3 com $H = 5$ e TE1 com $H = 3$, respectivamente, as quais correspondem a gaps de 64.69 e 77.44, respectivamente.

Os problemas com custos euclideanos e raíz no extremo da grelha com 60 nodos destino são muito difíceis de resolver e por isso não foi possível determinar o valor óptimo nem um seu limite inferior, através da relaxação lagrangeana, das instâncias de teste TE2 e TE3 com $H = 4$ e $H = 5$. Os gaps, da solução fornecida pela heurística e para estas instâncias, foram calculados utilizando como limite inferior o valor da solução dual ascendente. Assinalámos este facto colocando o símbolo * junto do valor do gap. Sabemos que estes limites inferiores são "fracos"o que justifica os elevados valores dos gaps que obtivemos nestes casos, notamos que, o menor destes gaps é de 466.51 obtido para a instância TE2 com $H = 5$. Por este motivo optámos por excluir da análise que fazemos estes resultados. Assim, analisando os resultados obtidos com a aplicação da heurística notamos que o menor valor entre o valor da solução obtida pela heurística e o valor óptimo, que corresponde ao melhor limite superior, é de 1988 unidades, ao qual corresponde um gap de 135.89 e

foi obtido para a instância TE2 com $H = 3$. À excepção deste valor, os restantes valores obtidos com a aplicação da heurística são os piores resultados obtidos de entre todas as instâncias testadas. Notamos que o pior limite superior encontrado corresponde à instância TE1 com $H = 3$. Neste caso o gap é de 218.36 e corresponde à diferença máxima de 3330 unidades.

PROB	H	OPT	Dual Ascendente			Heurística		
			valor	*tempo*	*gap*	*valor*	*tempo*	*gap*
	3	274	118	0.03	56.93	325	0.04	18.61
TR1,60	4	207	113	0.01	45.41	253	0.04	22.22
	5	189	115	0.05	39.15	245	0.05	29.63
	3	243	73	0.04	69.96	266	0.04	9.47
TR2,60	4	184	81	0.04	55.98	202	0.04	9.78
	5	132	90	0.03	31.82	148	0.06	12.12
	3	288	91	0.07	68.40	371	0.07	28.82
TR3,60	4	227	72	0.03	68.28	325	0.03	43.17
	5	206	81	0.06	60.68	286	0.06	38.83
	3	204	92	0.05	54.90	251	0.05	23.04
TR4,60	4	163	90	0.02	44.79	184	0.02	12.88
	5	151	95	0.03	37.09	181	0.05	19.87
	3	230	97	0.03	57.03	295	0.03	28.26
TR5,60	4	180	115	0.03	36.11	212	0.03	17.78
	5	170	102	0.04	40.00	211	0.08	24.12

Tabela 5.9: Resultados computacionais para instâncias com custos aleatórios e com 60 nodos destino

Para as instâncias com custos aleatórios com 60 nodos destino (ver Tabela 5.9) os melhores limites inferior e superior foram obtidos, respectivamente, para as instâncias TR2 com $H = 5$ e TR2 com $H = 3$, sendo as diferenças entre o valor óptimo e o valor da solução encontrada de 42 e 23 unidades, respectivamente, a que correspondem gaps de 31.82 e 9.47. Notamos ainda que estes foram os melhores limites encontrados para as instâncias com 60 nodos destino. No entanto, para a heurística, este valor não corresponde à menor diferença observada. Foi para a instância TR2 com $H = 5$ que se verificou a diferença mínima de 16 unidades correspondendo a um gap de 12.12. A maior diferença entre o valor óptimo e o valor da solução dual ascendente é de 197 unidades a que corresponde um gap de 68.40 obtida para a instância TR3 com $H = 3$. Tendo sido, no entanto, o pior limite inferior

registado para a instância TR2 com $H = 3$. Neste caso o gap é de 69.96, que corresponde a uma diferença de 170 unidades. No que diz respeito à heurística, o pior limite inferior verificou-se para a instância TR3 com $H = 4$, sendo neste caso o valor do gap de 43.17 o qual corresponde a uma diferença entre o valor da solução obtida pela heurística e o valor óptimo de 98 unidades, sendo esta a diferença máxima registada.

As relações que se verificaram para as instâncias com 20 e 40 nodos destino também se verificam para as instâncias com 60 nodos destino. Analisando os resultados obtidos com a aplicação do algoritmo Dual Ascendente notamos que os problemas com custos euclideanos e raíz no extremo da grelha (TE) são os que apresentam os maiores valores dos gaps sendo os menores os valores obtidos para os problemas euclideanos com raíz no centro da grelha (TC). Para a heurística os maiores valores dos gaps verificam-se também nos problemas com custos euclideanos e raíz no extremo da grelha (TE) e os menores valores dos gaps dizem respeito aos problemas com custos aleatórios (TR).

Comparando os resultados do algoritmo Dual Ascendente com os da heurística, notamos também a mesma relação, isto é os valores dos gaps dos problemas com custos euclideanos (TC e TE) são normalmente superiores para a heurística e os valores dos gaps para os problemas com custos aleatórios (TR) são normalmente superiores para o algoritmo Dual Ascendente.

Assim, de um modo geral, podemos dizer que os problemas com custos euclideanos e raíz no extremo da grelha (TE) são os de mais difícil resolução. Notamos que os piores resultados foram obtidos com este tipo de instâncias, quer para o algoritmo Dual Ascendente quer para a heurística, tendo sido na heurística que se verificaram os maiores valores dos gaps. No geral os problemas com custos euclideanos e raíz no centro da grelha (TC) são os que apresentam menores valores dos gaps para o algoritmo Dual Ascendente e os problemas com custos aleatórios (TR) os que apresentam menores valores para a heurística.

Podemos dizer que, no geral, comparando os limites inferiores e superiores obtidos para cada problema, o algoritmo dual ascendente fornece melhor solução do que a heurística para os problemas com custos euclideanos (TC e TE) e que a relação contrária se verifica para as instâncias com custos aleatórios (TR).

Notamos ainda que, à medida que a dimensão do problema aumenta os valores dos gaps (quer do algoritmo Dual Ascendente quer da heurística) aumentam também, o que significa que comparativamente as instâncias de maior dimensão apresentam piores resultados.

Muitos dos resultados apresentados por Gouveia[10] foram obtidos usando relaxações lineares de modelos que envolvem um elevado número de variáveis e restrições, pelo que na tentativa de reduzir o tamanho das instâncias foi usado o seguinte teste.

> **Teste de eliminação de arcos:** Se $c_{ij} > c_{0j}$, então nenhuma solução óptima inclui o arco (i, j). Se $c_{ij} = c_{0j}$, então existe uma solução óptima que não inclui o arco (i, j) $(i \neq 0)$.

Notamos que se a solução inclui tais arcos então uma outra solução admissível com o mesmo custo (no caso de $c_{ij} = c_{0j}$) ou uma solução admissível melhor (no caso de $c_{ij} > c_{0j}$) pode ser obtida se removermos o arco (i, j) e incluirmos o arco $(0, j)$. Deste modo, o arco (i, j) pode ser eliminado do problema permitindo, como consequência eliminar também algumas restrições.

Este teste foi também aplicado a todas as instâncias do problema e o número de arcos que foi possível eliminar em cada uma das instâncias testadas encontra-se nas Tabelas 5.10, 5.11 e 5.12. Na primeira linha de cada tabela indica-se o número de nodos destino do problema (n) e o número de arcos do correspondente grafo orientado ($|A|$). Escreveram-se os problemas em coluna e as linhas dizem respeito ao número de arcos que foi possível eliminar (nº), e à percentagem de arcos eliminados (%) tendo por base o número total de arcos da instância testada e que está indicado na primeira linha da tabela ($|A|$).

Para as instâncias com 20 nodos destino (ver Tabela 5.10) o número de arcos eliminados varia entre 72 para a instância TE1 e 273 para as instâncias TC4 e TC5, correspondendo a uma percentagem de 18.00 e 68.25, respectivamente. Notamos que o número de arcos eliminados é consideravelmente superior para os problemas euclideanos com raíz no centro da grelha (TC) e menor para os problemas euclideanos com raíz no extremo da grelha (TE).

n=20, $|A| = 400$

			PROB		
arcos eliminados	TC1,20	TC2,20	TC3,20	TC4,20	TC5,20
nº	247	255	244	273	273
%	61.75	63.75	61.00	68.25	68.25
			PROB		
arcos eliminados	TE1,20	TE2,20	TE3,20	TE4,20	TE5,20
nº	72	106	107	90	124
%	18.00	26.50	26.75	22.50	31.00
			PROB		
arcos eliminados	TR1,20	TR2,20	TR3,20	TR4,20	TR5,20
nº	191	194	202	172	171
%	47.75	48.50	50.50	43.00	42.75

Tabela 5.10: Arcos que foi possível eliminar nas instâncias com 20 nodos destino

Também para as instâncias com 40 nodos destino (ver Tabela 5.11) se verifica que o número de arcos eliminados é consideravelmente superior para os problemas euclideanos com raíz no centro da grelha (TC) e menor para os problemas euclideanos com raíz no extremo da grelha (TE). Notamos que para estes problemas a percentagem de arcos eliminados varia entre 23.06 para a instância TE3 e 67.81 para a instância TC5, correspondendo a 369 e 1085 arcos eliminados, respectivamente.

As conclusões apresentadas para as instâncias com 20 e 40 nodos destino também são válidas para as instâncias com 60 nodos destino (ver Tabela 5.12), isto é, o número de arcos eliminados é consideravelmente superior para os problemas euclideanos com raíz no centro da grelha (TC) e bastante menores para os problemas euclideanos com raíz no extremo da grelha (TE). Notamos que, de entre as instâncias com 60 nodos destino, é para a instância TE2 que se observou a menor redução sendo, neste caso, o número de arcos eliminados igual a 822 a que corresponde uma percentagem de 22.83. A maior redução verificou-se para a instância TC1, tendo-se neste caso eliminado 2501 arcos que corresponde a 69.47% dos arcos do problema.

Analisando estes resultados no seu conjunto, concluímos que foi possível eliminar mais arcos para os problemas com custos euclideanos e raíz no centro da grelha (TC). Para estes

n=40, $|A| = 1600$

			PROB		
arcos eliminados	TC1,40	TC2,40	TC3,40	TC4,40	TC5,40
nº	1068	1094	1067	1080	1085
%	66.75	63.38	66.69	67.50	67.81
			PROB		
arcos eliminados	TE1,40	TE2,40	TE3,40	TE4,40	TE5,40
nº	404	447	369	432	404
%	25.25	29.94	23.06	27.00	25.25
			PROB		
arcos eliminados	TR1,40	TR2,40	TR3,40	TR4,40	TR5,40
nº	789	781	605	795	921
%	49.31	48.81	37.81	49.69	57.56

Tabela 5.11: Arcos que foi possível eliminar nas instâncias com 40 nodos destino

problemas a percentagem de arcos eliminados varia entre 61.00 para a instância TC3,20 e 69.47 para a instância TC1,60. A percentagem de arcos eliminados é bastante menor para os problemas com custos euclideanos e raíz no extremo da grelha (TE) e varia entre 18.00 para a instância TE1,20 e 31.00 para a instância TE5,20. O que quer dizer que estes últimos (TE) são, em geral, problemas mais densos.

Após eliminarmos os arcos com base neste teste, reaplicámos o algoritmo Dual Ascendente. Notamos, tal como esperado, que não houve alterações significativas nos tempos computacionais obtidos, apenas se verificou uma ligeira redução em algumas das instâncias testadas.

n=60, $|A| = 3600$

			PROB		
arcos eliminados	TC1,60	TC2,60	TC3,60	TC4,60	TC5,60
nº	2501	2436	2489	2499	2475
%	69.47	67.67	69.14	69.42	68.75

			PROB
arcos eliminados	TE1,60	TE2,60	TE3,60
nº	923	822	827
%	25.64	22.83	22.97

			PROB		
arcos eliminados	TR1,60	TR2,60	TR3,60	TR4,60	TR5,60
nº	1517	1780	1525	1743	1747
%	42.14	49.44	42.36	48.42	48.53

Tabela 5.12: Arcos que foi possível eliminar nas instâncias com 60 nodos destino

Capítulo 6

Considerações Finais

Neste trabalho apresentamos um algoritmo Dual Ascendente para o problema da Árvore de Suporte de Custo Mínimo com Restrições de Salto. Apresentamos duas formulações de fluxos orientadas para este problema. A primeira obtém-se adicionando-se as restrições de salto a uma conhecida formulação de fluxos para o problema da Árvore de Suporte de Custo Mínimo e a segunda é uma formulação alternativa mais compacta e foi obtida por Gouveia[11] utilizando a técnica de redefinição de variáveis de Martin[16]. Utilizámos a formulação dual linear da formulação alternativa e aplicámos a técnica Dual Ascendente para desenvolver um algoritmo Dual Ascendente com a finalidade de obter um limite inferior para o valor óptimo do problema HMST. Descrevemos ainda uma heurística baseada na solução dual ascendente para obter um limite superior para o valor óptimo do problema. Apresentámos resultados computacionais para instâncias do problema correspondentes a grafos com 20,40 e 60 nodos destino e para valores de H de 3,4 e 5.

Os resultados obtidos mostram que apesar do Algoritmo Dual Ascendente e da heurística baseada na solução dual ascendente determinarem de forma muito rápida, respectivamente, um limite inferior e um limite superior para o valor óptimo do problema estes limites estão "longe"do valor óptimo do problema.

Para finalizar este trabalho deixamos uma observação que poderá vir a ser objecto de estudo no futuro.

Quando definimos o algoritmo dual ascendente, não definimos nenhuma regra para a

escolha de qual o nodo a seleccionar em cada iteração nem nenhuma regra para a escolha do arco a incluir na solução em caso de empate. A escolha que fazemos influência a solução dual ascendente que se encontra, por isso consideramos que seria interessante estudar e comparar diferentes regras de selecção do nodo a considerar em cada iteração e diferentes regras de selecção de arcos a incluir na solução em caso de empate.

Bibliografia

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall, 1993.

[2] A. Balakrishnan, T.L. Magnanti, and P. Mirchandani. A Dual-Based Algorithm for Multi-Level Network Design. *Management Science*, 40:567–581, 1994.

[3] A. Balakrishnan, T.L. Magnanti, and R.T. Wong. A Dual-Ascent Procedure for Large-Scale Uncapacitated Network Design. *Operations Research*, 37(5):716–740, 1989.

[4] G. Dahl. The 2-Hop Spanning Tree Problem. *Operations Research Letters*, 23:21–26, 1998.

[5] G. Dahl, L. Gouveia, and C. Requejo. On Formulations and Methods for the Hop-Constrained Minimum Spanning Tree Problem. In M.G.C. Resend and P.M. Pardalos, editors, *Handbook of optimization in telecommunications*. Spinger Science + Business Media, 2006.

[6] D. Erlenkotter. A Dual Based Procedure for Uncapacitated Facility Location. *Operations Research*, 26:992–1009, 1978.

[7] A. Frangioni and G. Gallo. A Bundle Type Dual-Ascent Approach to Linear Multicommodity Min-Cost Flow Problems. *INFORMS Journal on Computing*, 11:370–393, 1999.

[8] B. Gendron. A Note on "a Dual-Ascent Approach to the Fixed-charge Capacitated Network Design Problem". *European Journal of Operational Research*, 138:671–675, 2002.

[9] L. Gouveia. Using the Miller-Tucker-Zemlin Constraints to Formulate a Minimal Spanning Tree Problem with Hop Constraints. *Computers and Operations Research*, 22(9):959–970, 1995.

[10] L. Gouveia. Multicommodity Flow Models for Spanning Trees with Hop Constraints. *European Journal of Operational Research*, 95:178–190, 1996.

[11] L. Gouveia. Using Variable Redefinition for Computing Lower Bounds for Minimum Spanning and Steiner Trees with Hop Constraints. *INFORMS Journal on Computing*, 10(2):180–188, 1998.

[12] L. Gouveia and C. Requejo. A New Lagrangean Relaxation Approach for the Hop-Constrained Minimum Spanning Tree Problem. *European Journal of Operational Research*, 132(3):539–552, 2001.

[13] J.W. Herrmann, G. Ioannou, I. Minis, and J.M. Proth. A Dual Ascent Approach to the Fixed-Charge Capacitated Network Design Problem. *European Journal of Operational Research*, 95:476–490, 1996.

[14] T. Magnanti and L. Wolsey. Optimal Trees. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Models, Handbooks in Operations Research and Management Science, Vol. 7*. Elsevier Science Publishers, North-Holland, 1995.

[15] P. Manyem and M. Stallmann. Some Approximation Results in Multicasting. Working paper, North Carolina State University, 1996.

[16] R.K. Martin. Generating Alternative Mixed-Integer Programming Models Using Variable Redefinition. *Operations Research*, 35(6):820–831, 1987.

[17] S. Nguyen, S. Pallottino, and M.G. Scutella. A New Dual Algorithm for the Shortest Path Reoptimization. Working paper, Università di Pisa, Italy, 1999.

[18] S. Pallottino and M.G. Scutella. Dual Algorithms for the Shortest Path Tree Problem. *Networks*, 29:125–133, 1997.

[19] E. Rosenberg. Dual Ascent for Uncapacitated Telecommunications Network Design with Access, Backbone, and Switch Costs. *Telecommunication Systems*, 4:423–435, 2001.

[20] R.T. Wong. A Dual Ascent Approach for Steiner Tree Problems on a Directed Graph. *Mathematical Programming*, 28:271–287, 1984.

[21] K.A. Woolston and S.L. Albin. The Design of Centralized Networks with Reliability and Availability Constraints. *Computers and Operations Research*, 15(3):207–217, 1988.